```
ASM80 GETF.ASM DEBUG


ISIS-II 8080/8085 MACRO ASSEMBLER, V2.0        GETF     PAGE    1
GETF  - CP/M 3.0 Input Redirection - August 1982

  LOC  OBJ        SEQ        SOURCE STATEMENT

                    1 $title('GETF  - CP/M 3.0 Input Redirection - August 1982')
                    2         name    getf
                    3 ;*****************************************************************
                    4 ;
                    5 ;       get  'Input Redirection Initializer'  version 3.0
                    6 ;
                    7 ;       11/30/82 - Doug Huskey
                    8 ;*****************************************************************
                    9 ;
                   10 ;
                   11 ;       Copyright (c) 1982
                   12 ;       Digital Research
                   13 ;       P.O. Box 579
                   14 ;       Pacific Grove, Ca.
                   15 ;       93950
                   16 ;
                   17 ;
                   18 ;       generation procedure
                   19 ;
                   20 ;       seteof get.plm
                   21 ;       seteof getscan.dcl
                   22 ;       seteof getf.asm
                   23 ;       seteof getscan.plm
                   24 ;       seteof parse.asm
                   25 ;       is14
                   26 ;       asm80 getf.asm debug
                   27 ;       asm80 mcd80a.asm debug
                   28 ;       asm80 parse.asm debug
                   29 ;       plm80 get.plm pagewidth(100) debug optimize
                   30 ;       link mcd80a.obj,get.obj,parse.obj,getf.obj,plm80.lib to get.mod
                   31 ;       locate get.mod code(0100H) stacksize(100)
                   32 ;       era get.mod
                   33 ;       cpm
                   34 ;       objcpm get
                   35 ;       rmac getrsx
                   36 ;       link getrsx[op]
                   37 ;       era get.rsx
                   38 ;       ren get.rsx=getrsx.prl
                   39 ;       gencom get.com
                   40 ;       gencom get.com get.rsx
                   41 ;
                   42 ;
                   43 ;
                   44 ;       This module is called as an external routine by the
                   45 ;       PL/M routines GET and SUBMIT.  It is passed a structure
                   46 ;       with the following format:
                   47 ;
                   48 ;
                   49 ;                 declare getpb structure
                   50 ;                   (input$type   byte,
                   51 ;                    echo$flag    byte,
                   52 ;                    filtered$flag byte,
```

CP/M - PLUS V3.0
COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. # CP3-135

```
LOC  OBJ        SEQ        SOURCE STATEMENT

                  53 ;                    program$flag  byte);
                  54 ;
                  55 ;       input$type      = 0     > console input (default)
                  56 ;                       = 1     > auxiliary output
                  57 ;
                  58 ;       echo            = true  > echo input to real device
                  59 ;                                 (default)
                  60 ;                       = false > don't echo input (output is
                  61 ;                                 still echoed)
                  62 ;       filtered        = true  > convert control characters
                  63 ;                                 to a printable form
                  64 ;                                 preceeded by an ^ in echo
                  65 ;                                 (default)
                  66 ;                       = false > no character conversions
                  67 ;       program         = false > continue until EOF or
                  68 ;                                 GET INPUT FROM CONSOLE
                  69 ;                                 command
                  70 ;                       = true  > active only until program
                  71 ;                                 termination
                  72 ;
                  73         public  getf
                  74         extrn   moni,fcb,memsiz
                  75 ;
                  76 ;
FFFF              77 true            equ     0ffffh
0000              78 false           equ     00000h
                  79 ;
FFFF              80 biosfunctions   equ     true            ;intercept BIOS conin & constat
                  81 ;
                  82 ;
                  83 ;       low memory locations
                  84 ;
0000              85 wboot   equ     0000h
0001              86 wboota  equ     wboot+1
                  87 ;
                  88 ;       equates for non graphic characters
                  89 ;
000D              90 cr      equ     0dh     ; carriage return
000A              91 lf      equ     0ah     ; line feed
                  92 ;
                  93 ;       BDOS function equates
                  94 ;
0001              95 cinf    equ     1       ;read character
0002              96 coutf   equ     2       ;output character
0006              97 crawf   equ     6       ;raw console I/O
000A              98 creadf  equ     10      ;read buffer
000B              99 cstatf  equ     11      ;status
0005             100 pchrf   equ     5       ;print character
0009             101 pbuff   equ     9       ;print buffer
000F             102 openf   equ     15      ;open file
0010             103 closef  equ     16      ;close file
0013             104 delf    equ     19      ;delete file
0014             105 dreadf  equ     20      ;disk read
001A             106 dmaf    equ     26      ;set dma function
0019             107 curdrv  equ     25
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
LOC  OBJ          SEQ          SOURCE STATEMENT

0020              108 userf    equ     32      ;set/get user number
0031              109 scbf     equ     49      ;set/get system control block word
003C              110 rsxf     equ     60      ;RSX function call
0080              111 initf    equ     128     ;GET initialization sub-function no.
0081              112 killf    equ     129     ;GET delete sub-function no.
008D              113 jkillf   equ     141     ;JOURNAL delete sub-function no.
                  114 ;
                  115 ;       System Control Block definitions
                  116 ;
003A              117 scba     equ     03ah    ;offset of scbadr from SCB base
00B4              118 ccpflg2  equ     0b4h    ;offset of 2nd ccp flag byte from pg bound
00AA              119 errflg   equ     0aah    ;offset of error flag from page boundary
00CF              120 conmode  equ     0cfh    ;offset of console mode from page boundary
00D4              121 listcp   equ     0d4h    ;offset of ^P flag from page boundary
00F9              122 common   equ     0f9h    ;offset of common memory base from pg. bound
0068              123 wbootfx  equ     068h    ;offset of warm boot jmp from page. bound
006E              124 constfx  equ     06eh    ;offset of constat jmp from page. bound
0074              125 coninfx  equ     074h    ;offset of conin jmp from page. bound
007A              126 conoufx  equ     07ah    ;offset of conout jmp from page. bound
0080              127 listfx   equ     080h    ;offset of list jmp from page. bound
0098              128 realdos  equ     098h    ;offset of real BDOS entry from pg. bound
                  129 ;
                  130 ;       Restore mode equates    (used with inr a, rz, rm, rpe, ret)
                  131 ;
00FF              132 norestore        equ     0ffh    ;no BIOS interception
007F              133 biosonly         equ     07fh    ;restore BIOS jump table only
0080              134 stfix    equ     080h    ;restore BIOS jump table and
                  135                                  ;restore JMP in RESBDOS for constat
0000              136 everything       equ     0       ;restore BIOS jump table and jmps in
                  137                                  ;RESBDOS (default mode)
                  138 ;
                  139 ;       Instructions
                  140 ;
0021              141 lxih             equ     21h     ;LXI H, instruction
00C3              142 jmpi             equ     0c3h    ;JMP instruction
0022              143 shldi            equ     22h     ;SHLD instruction
                  144 ;
                  145 ;*****************************************************************
                  146 ;               START OF INITIALIZATION CODE
                  147 ;*****************************************************************
                  148
                  149                  cseg
                  150
                  151 getf:
                  152         ;get parameters
0000 60           153         mov     h,b
0001 69           154         mov     l,c             ;HL = .(parameter block)
0002 7E           155         mov     a,m             ;input type 0=con:,1=aux:
0003 FE01         156         cpi     1               ;is it aux?
0005 CA2301    C  157         jz      notimp          ;error if so
0008 23           158         inx     h
0009 7E           159         mov     a,m             ;echo/noecho mode
000A 32ED01    C  160         sta     echo
000D 23           161         inx     h
000E 7E           162         mov     a,m             ;cooked/raw mode
```

```
 LOC  OBJ          SEQ         SOURCE STATEMENT

000F 32EE01   C    163         sta     cooked
0012 23            164         inx     h
0013 7E            165         mov     a,m
0014 32EF01   C    166         sta     program
                   167         ;
                   168         ;check if enough memory
                   169         ;
0017 2A0000   E    170         lhld    memsiz
001A 7C            171         mov     a,h
001B FE20          172         cpi     20h
001D DA3E01   C    173         jc      nomem
                   174         ;
                   175         ;close to get those blocks in the directory
                   176         ;
0020 110000   E    177         lxi     d,fcb
0023 0E10          178         mvi     c,closef
0025 CD0000   E    179         call    mon1
                   180         ;
                   181         ;check if drive specified
0028 210000   E    182         lxi     h,fcb
002B 7E            183         mov     a,m             ;drive code
002C B7            184         ora     a               ;default?
002D C23900   C    185         jnz     movfcb
                   186         ;
                   187         ;set to current drive, if  not
                   188         ;
0030 E5            189         push    h               ;save .fcb
0031 0E19          190         mvi     c,curdrv
0033 CD0000   E    191         call    mon1
0036 E1            192         pop     h               ;a=current drive, hl=.fcb
0037 3C            193         inr     a
0038 77            194         mov     m,a             ;set fcb to force drive select
                   195         ;
                   196 movfcb: ;copy default fcb up into data area for move to RSX
                   197         ;
0039 11F101   C    198         lxi     d,subfcb
003C 012000        199         lxi     b,32            ;length of fcb
003F CD5601   C    200         call    ldir            ;move it to subfcb
                   201         ;
                   202         ;initialize other variables to be moved to RSX
                   203         ;
0042 CD4401   C    204         call    getusr          ;get current user number
0045 32F001   C    205         sta     subusr          ;save for redirection file I/O
0048 CD4C01   C    206         call    getscbadr
004B 22DB01   C    207         shld    scbadr          ;System Control Block address
                   208         ;
                   209         ;get real BDOS address (bypass chain to check for user break)
                   210         ;
004E 2E98          211         mvi     l,realdos
0050 5E            212         mov     e,m
0051 23            213         inx     h
0052 56            214         mov     d,m
0053 EB            215         xchg
0054 22EB01   C    216         shld    realbdos+1
                   217         ;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
LOC  OBJ          SEQ         SOURCE STATEMENT

                  218         ;check for user abort
                  219         ;
0057 EB           220         xchg
0058 2ECF         221         mvi     l,conmode
005A 7E           222         mov     a,m
005B F601         223         ori     1               ;set ^C status mode
005D 77           224         mov     m,a
005E 0E0B         225         mvi     c,cstatf
0060 CDEA01    C  226         call    realbdos        ;check for user abort
0063 B7           227         ora     a
0064 C22B01    C  228         jnz     error1          ;abort if so
                  229         ;
                  230         ;get address of initialization table in RSX
                  231         ;
0067 0E3C         232         mvi     c,rsxf
0069 116101    C  233         lxi     d,journkill
006C CD0000    E  234         call    mon1            ;terminate any PUT INPUT commands
006F 0E3C         235         mvi     c,rsxf
0071 116201    C  236         lxi     d,rsxinit
0074 CD0000    E  237         call    mon1            ;call GET.RSX initialization routine
0077 E5           238         push    h               ;save for move at end of setup
0078 5E           239         mov     e,m
0079 23           240         inx     h
007A 56           241         mov     d,m             ;DE = .RSXKILL flag
007B D5           242         push    d               ;set flag to zero if successfull
007C 23           243         inx     h               ;HL = .(real bios status routine)
007D E5           244         push    h
                  245         ;
                  246 if biosfunctions
                  247         ;
                  248         ;check if BIOS jump table looks valid (jmp in right places)
007E 2A0100       249         lhld    wboota
0081 110300       250         lxi     d,3
0084 19           251         dad     d               ;HL = .(jmp constat address)
0085 7E           252         mov     a,m
0086 FEC3         253         cpi     jmpi            ;should be a jump
0088 C21201    C  254         jnz     bioserr         ;skip bios redirection if not
008B 19           255         dad     d               ;HL = .(jmp conin address)
008C 7E           256         mov     a,m
008D FEC3         257         cpi     jmpi
008F C21201    C  258         jnz     bioserr         ;skip bios redirection if not
                  259         ;
                  260         ;fix up RESBDOS to do BIOS calls to intercepted functions
                  261         ;
0092 2ADB01    C  262         lhld    scbadr
0095 2EFA         263         mvi     l,common+1
0097 7E           264         mov     a,m             ;get high byte of common base
0098 B7           265         ora     a
0099 C2A400    C  266         jnz     fix0            ;high byte = zero if non-banked
009C 3E7F         267         mvi     a,biosonly
009E 32E101    C  268         sta     biosmode
00A1 C3D000    C  269         jmp     trap            ;skip code that fixes resbdos
                  270         ;fix BIOS constat
00A4 2E6E         271 fix0:   mvi     l,constfx               ;hl = .constfx in SCB
00A6 7E           272         mov     a,m
```

```
 LOC  OBJ        SEQ         SOURCE STATEMENT

 00A7 FEC3       273         cpi     jmpi            ;is it a jump instruction?
 00A9 CAB100  C  274         jz      fix1            ;jump if so
 00AC 3E7F       275         mvi     a,biosonly      ;whoops already changed
 00AE 32E101  C  276         sta     biosmode        ;restore jump table only
 00B1 3621       277 fix1:   mvi     m,lxih
                 278         ;fix BIOS conin
 00B3 2E74       279         mvi     l,coninfx       ;hl = .coninfx in SCB
 00B5 7E         280         mov     a,m
 00B6 FEC3       281         cpi     jmpi            ;is it a jump instruction?
 00B8 3AE101  C  282         lda     biosmode
 00BB CAC400  C  283         jz      fix2            ;jump if so
 00BE FE7F       284         cpi     biosonly
 00C0 C21201  C  285         jnz     bioserr         ;error if conin is LXI but not constat
 00C3 AF         286         xra     a               ;zero accumulator to jnz below
 00C4 FE7F       287 fix2:   cpi     biosonly        ;was const already an LXI h?
 00C6 C2CE00  C  288         jnz     fix3            ;jmp if not
 00C9 3E80       289         mvi     a,stfix ;restore constat jmp but not conin
 00CB 32E101  C  290         sta     biosmode
 00CE 3621       291 fix3:   mvi     m,lxih
                 292         ;get addresses of RSX const and conin traps
 00D0 E1         293 trap:   pop     h
 00D1 4E         294         mov     c,m             ;HL = .(.bios constat trap)
 00D2 23         295         inx     h
 00D3 46         296         mov     b,m             ;BC = .bios constat trap in RSX
 00D4 23         297         inx     h
 00D5 E5         298         push    h               ;save for CONIN setup
                 299         ;
                 300         ;patch RSX constat entry into BIOS jump table
                 301         ;save real constat address in RSX exit table
                 302         ;
 00D6 2A0100     303         lhld    wboota
 00D9 110400     304         lxi     d,4
 00DC 19         305         dad     d               ;HL = .(jmp constat address)
 00DD 22E301  C  306         shld    constjmp        ;save for RSX restore at end
 00E0 5E         307         mov     e,m
 00E1 71         308         mov     m,c
 00E2 23         309         inx     h
 00E3 56         310         mov     d,m             ;DE = constat address
 00E4 70         311         mov     m,b             ;BIOS constat jumps to RSX
 00E5 EB         312         xchg
 00E6 22DD01  C  313         shld    biosta          ;save real constat address
                 314         ;
                 315         ;get address of RSX bios conin entry point
                 316         ;
 00E9 E1         317         pop     h               ;HL = .(RSX BIOS conin trap)
 00EA 4E         318         mov     c,m
 00EB 23         319         inx     h
 00EC 46         320         mov     b,m
                 321         ;
                 322         ;patch RSX conin entry into BIOS jump table
                 323         ;save real conin address in RSX exit table
                 324         ;
 00ED EB         325         xchg
 00EE 23         326         inx     h               ;past jmp instruction
 00EF 23         327         inx     h               ;HL = .(conin address)
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
 LOC  OBJ          SEQ          SOURCE STATEMENT

 00F0 22E701   C   328          shld    coninjmp
 00F3 5E           329          mov     e,m
 00F4 71           330          mov     m,c
 00F5 23           331          inx     h
 00F6 56           332          mov     d,m             ;DE = conin address
 00F7 70           333          mov     m,b             ;BIOS conin jumps to RSX
 00F8 EB           334          xchg
 00F9 22DF01   C   335          shld    biosin          ;save real conin address
                   336 endif
                   337          ;
                   338          ;move data area to RSX
                   339          ;
                   340 rsxmov:
 00FC E1           341          pop     h               ;HL = .Kill flag in RSX
 00FD 34           342          inr     m               ;switch from FF to 0
 00FE 21DB01   C   343          lxi     h,movstart
 0101 D1           344          pop     d               ;RSX data area address
 0102 013700       345          lxi     b,movend-movstart
 0105 CD5601   C   346          call    ldir
 0108 0E06         347          mvi     c,crawf
 010A 1EFD         348          mvi     e,0fdh          ;raw console input
 010C CD0000   E   349          call    moni            ;prime RSX by reading a char
 010F C30000       350          jmp     wboot
                   351
                   352 if biosfunctions
                   353 ;
                   354 ;       can't do BIOS redirection
                   355 ;
                   356 bioserr:
 0112 116301   C   357          lxi     d,nobios
 0115 0E09         358          mvi     c,pbuff
 0117 CD0000   E   359          call    moni
 011A 21E101   C   360          lxi     h,biosmode
 011D 36FF         361          mvi     m,norestore     ;no bios redirection
 011F E1           362          pop     h               ;throw away bios constat trap adr
 0120 C3FC00   C   363          jmp     rsxmov
                   364 endif
                   365 ;
                   366 ;       auxiliary redirection
                   367 ;
                   368 notimp:
 0123 118801   C   369          lxi     d,notdone
                   370 error:
 0126 0E09         371          mvi     c,pbuff
 0128 CD0000   E   372          call    moni
 012B 0E10         373 error1:  mvi     c,closef
 012D 110000   E   374          lxi     d,fcb
 0130 CD0000   E   375          call    moni
 0133 0E13         376          mvi     c,delf
 0135 110000   E   377          lxi     d,fcb
 0138 CD0000   E   378          call    moni
 013B C30000       379          jmp     wboot
                   380 ;
                   381 ;       insufficient memory
                   382 ;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
LOC  OBJ          SEQ         SOURCE STATEMENT

013E 11BE01   C    383 nomem:  lxi     d,memerr
0141 C32601   C    384         jmp     error
                   385
                   386 ;
                   387 ;       get/set user number
                   388 ;
0144 3EFF          389 getusr: mvi     a,0ffh          ;get current user number
0146 5F            390 setusr: mov     e,a             ;set current user number (in A)
0147 0E20          391         mvi     c,userf
0149 C30000   E    392         jmp     mon1
                   393 ;
                   394 ;       get system control block address
                   395 ;       (BDOS function #49)
                   396 ;
                   397 ;       exit:   hl = system control block address
                   398 ;
                   399 getscbadr:
014C 0E31          400         mvi     c,scbf
014E 115401   C    401         lxi     d,data49
0151 C30000   E    402         jmp     mon1
                   403 ;
0154 3A            404 data49: db      scba,0          ;data structure for getscbadd
0155 00
                   405 ;
                   406 ;       copy memory bytes (emulates z80 ldir instruction)
                   407 ;
0156 7E            408 ldir:   mov     a,m             ;get byte
0157 12            409         stax    d               ;store it at destination
0158 23            410         inx     h               ;advance pointers
0159 13            411         inx     d
015A 0B            412         dcx     b               ;decrement byte count
015B 79            413         mov     a,c             ;loop if non-zero
015C B0            414         ora     b
015D C25601   C    415         jnz     ldir
0160 C9            416         ret
                   417 ;
                   418 ;***************************************************************
                   419 ;               DATA AREA
                   420 ;***************************************************************
                   421
                   422         ;
0161 8D            423 journkill:      db      jkillf
0162 80            424 rsxinit:        db      initf
0163 5741524E      425 nobios:         db      'WARNING: Cannot redirect from BIOS',cr,lf,'$'
0167 494E473A
016B 2043616E
016F 6E6F7420
0173 72656469
0177 72656374
017B 2066726F
017F 6D204249
0183 4F53
0185 0D
0186 0A
0187 24
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
LOC  OBJ        SEQ        SOURCE STATEMENT

                426 notdone:
0188 4552524F   427        db 'ERROR: Auxiliary device redirection not implemented',cr,lf,'$'
018C 523A2041
0190 7578696C
0194 69617279
0198 20646576
019C 69636520
01A0 72656469
01A4 72656374
01A8 696F6E20
01AC 6E6F7420
01B0 696D706C
01B4 656D656E
01B8 746564
01BB 0D
01BC 0A
01BD 24
                428 memerr:
01BE 4552524F   429        db 'ERROR: Insufficient Memory',cr,lf,'$'
01C2 523A2049
01C6 6E737566
01CA 66696369
01CE 656E7420
01D2 4D656D6F
01D6 7279
01D8 0D
01D9 0A
01DA 24
                430        ;
                431 ;**************************************************************
                432 ;       Following variables are initialized by GET.COM
                433 ;       and moved to the GET RSX - Their order must not be changed
                434 ;**************************************************************
                435        ;
                436        ;
                437        ;
                438 movstart:
                439 inittable:                    ;addresses used by GET.COM for
01DB 0000       440 scbadr: dw      0            ;address of System Control Block
                441        ;
                442        if biosfunctions      ;GET.RSX initialization
                443        ;
01DD 0000       444 biosta: dw      0            ;set to real BIOS routine
01DF 0000       445 biosin: dw      0            ;set to real BIOS routine
                446        ;
                447                               ;restore only if changed when removed.
                448 biosmode:
01E1 00         449        db      0            ;if non-zero change LXI @jmpadr to JMP
                450                               ;when removed.
                451 restorebios:
                452        ;hl = real constat routine
                453        ;de = real conin routine
01E2 22         454        db      shldi
                455 constjmp:
01E3 0000       456        dw      0            ;address of const jmp initialized by COM
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
  LOC  OBJ          SEQ         SOURCE STATEMENT

  01E5 EB           457         xchg
  01E6 22           458         db      shldi
                    459 coninjmp:
  01E7 0000         460         dw      0           ;address of conin jmp initialized by COM
  01E9 C9           461         ret
                    462         endif
                    463         ;
                    464 realbdos:
  01EA C30000       465         jmp     0           ;address filled in by COM
                    466         ;
  01ED 01           467 echo:   db      1
  01EE 00           468 cooked: db      0
                    469         ;
                    470 program:
  01EF 00           471         db      0           ;true if only program input
  01F0 00           472 subusr: db      0           ;user number for redirection file
  01F1 01           473 subfcb: db      1           ;a:
  01F2 53595349     474         db      'SYSIN  '
  01F6 4E202020
  01FA 535542       475         db      'SUB'
  01FD 00           476         db      0,0
  01FE 00
  01FF 00           477 submod: db      0
  0200 00           478 subrc:  db      0
  0010              479         ds      16          ;map
  0211 00           480 subcr:  db      0
                    481         ;
                    482 movend:
                    483 ;*****************************************************************
                    484         end
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
PUBLIC SYMBOLS
GETF   C 0000

EXTERNAL SYMBOLS
FCB    E 0000    MEMSIZ E 0000    MON1   E 0000

USER SYMBOLS
BIOSER C 0112    BIOSFU A FFFF    BIOSIN C 01DF    BIOSMO C 01E1    BIOSON A 007F    BIOSTA C 01DB    CCPFLG A 00B4
CINF   A 0001    CLOSEF A 0010    COMMON A 00F9    CONINF A 0074    CONINJ C 01E7    CONMOD A 00CF    CONOUF A 007A
CONSTF A 006E    CONSTJ C 01E3    COOKED C 01EE    COUTF  A 0002    CR     A 000D    CRAWF  A 0006    CREADF A 000A
CSTATF A 000B    CURDRV A 0019    DATA49 C 0154    DELF   A 0013    DMAF   A 001A    DREADF A 0014    ECHO   C 01ED
ERRFLG A 00AA    ERROR  C 0126    ERROR1 C 012B    EVERYT A 0000    FALSE  A 0000    FCB    E 0000    FIX0   C 00A4
FIX1   C 00B1    FIX2   C 00C4    FIX3   C 00CE    GETF   C 0000    GETSCB C 014C    GETUSR C 0144    INITF  A 0080
INITTA C 01DB    JKILLF A 008D    JMPI   A 00C3    JOURNK C 0161    KILLF  A 0081    LDIR   C 0156    LF     A 000A
LISTCP A 00D4    LISTFX A 0080    LXIH   A 0021    MEMERR C 01BE    MEMSIZ E 0000    MON1   E 0000    MOVEND C 0212
MOVFCB C 0039    MOVSTA C 01DB    NOBIOS C 0163    NOMEM  C 013E    NOREST A 00FF    NOTDON C 0188    NOTIMP C 0123
OPENF  A 000F    PBUFF  A 0009    PCHRF  A 0005    PROGRA C 01EF    REALBD C 01EA    REALDO A 0098    RESTOR C 01E2
RSXF   A 003C    RSXINI C 0162    RSXMOV C 00FC    SCBA   A 003A    SCBADR C 01DB    SCBF   A 0031    SETUSR C 0146
SHLDI  A 0022    STFIX  A 0080    SUBCR  C 0211    SUBFCB C 01F1    SUBMOD C 01FF    SUBRC  C 0200    SUBUSR C 01F0
TRAP   C 00D0    TRUE   A FFFF    USERF  A 0020    WBOOT  A 0000    WBOOTA A 0001    WBOOTF A 0068

ASSEMBLY COMPLETE,  NO ERRORS
```

```
ISIS-II PL/M-80 V3.1 COMPILATION OF MODULE SUB
OBJECT MODULE PLACED IN SUBMIT.OBJ
COMPILER INVOKED BY:  PLM80 SUBMIT.PLM XREF PAGEWIDTH(100) DEBUG OPTIMIZE


            $ TITLE('CP/M 3.0 --- SUBMIT')
  1         sub:
            do;
            $include (copyrt.lit)
        =
        =   /*
        =     Copyright (C) 1982
        =     Digital Research
        =     P.O. Box 579
        =     Pacific Grove, CA 93950
        =   */
        =
            /*
              Revised:
                26 July 79  for CP/M 2.0
                01 July 82  for CP/M 3.0 by John Knight
                23 Aug  82  for CP/M 3.0 by Doug Huskey
                11 Sept 82  for CP/M 3.0 by Doug Huskey
                 1 Nov  82  for CP/M 3.0 by Doug Huskey

            */

            /*
                generation procedure

            seteof submit.plm
            seteof copyrt.lit
            isl4
            asm80 mcd80a.asm debug
            asm80 getf.asm debug
            asm80 parse.asm debug
            plm80 submit.plm pagewidth(100) debug optimize
            link mcd80a.obj,submit.obj,parse.obj,getf.obj,plm80.lib to submit.mod
            locate submit.mod code(0100H) stacksize(100)
            era submit.mod
            cpm
            objcpm submit
            rmac getrsx
            xref getrsx
            link getrsx[op]
            era get.rsx
            ren get.rsx=getrsx.prl
            gencom submit.com get.rsx

            */

  2  1      declare plm label public;


            /**********************************
            *                                 *
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
                /* B D O S   I N T E R F A C E *
                *                             *
                *******************************/

 3  1           declare
                   sfcb(33)  byte external,    /* default fcb */
                   buff(128) byte external;    /* default buffer */

 4  1             declare cmdrv     byte    external;  /* command drive      */
 5  1             declare fcb (1)   byte    external;  /* 1st default fcb    */
 6  1             declare fcb16 (1) byte    external;  /* 2nd default fcb    */
 7  1             declare pass0     address external;  /* 1st password ptr   */
 8  1             declare len0      byte    external;  /* 1st passwd length  */
 9  1             declare pass1     address external;  /* 2nd password ptr   */
10  1             declare len1      byte    external;  /* 2nd passwd length  */
11  1             declare tbuff (1) byte    external;  /* default dma buffer */

12  1           mon1: procedure(f,a) external;
13  2               declare f byte, a address;
                    /* bdos interface, no returned value */
14  2               end mon1;

15  1           mon2a: procedure(f,a) external;
16  2               declare f byte, a byte;
                    /* bdos interface, no returned value */
17  2               end mon2a;

18  1           mon2: procedure(f,a) byte external;
19  2               declare f byte, a address;
                    /* bdos interface, return byte value */
20  2               end mon2;

21  1           mon3: procedure(func,info) address external;
22  2               declare func byte;
23  2               declare info address;
24  2               end mon3;

25  1           parse:
                   procedure (pfcb) address external;
26  2                 declare pfcb address;
27  2           end parse;

28  1           getf:
                   procedure (input$type) external; /* does submit file processing */
29  2                 declare input$type address;
30  2           end getf;

                /*************************************
                *                                    *
                *        L I T E R A L S             *
                *                                    *
                *************************************/

31  1            declare lit literally 'literally',
                    dcl lit 'declare',
                    proc lit 'procedure',
                    addr lit 'address',
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
        ctll lit '0ch',
        lca  lit '110$0001b',  /* lower case a */
        lcz  lit '111$1010b',  /* lower case z */
        endfile lit '1ah',    /* cp/m end of file */
        sysin$endfile lit '0ffh',
        true literally '1',
        false literally '0',
        forever literally 'while true',
        cr literally '13',
        lf literally '10',
        what literally '63',
        temp$file$drive$offset literally '50h',
        con$type literally '0',
        cpmversion literally '30h',
        ctrli literally '09h';

      /****************************************
      *                                       *
      *   G L O B A L   V A R I A B L E S     *
      *                                       *
      ****************************************/

32  1   declare
        ln(9) byte initial('00001 : $'),
        ln1 byte at(.ln(0)),
        ln2 byte at(.ln(1)),
        ln3 byte at(.ln(2)),
        ln4 byte at(.ln(3)),
        ln5 byte at(.ln(4)),
        dfcb(36) byte initial(0,'SYSIN   $$$',0,0,0),
        drec byte at(.dfcb(32)),  /* current record */
        drrec address at(.dfcb(33)),  /* random record */
        drr2 byte at(.dfcb(35)),   /* random record byte 3 */
        dcnt byte,
        get$init$pb byte initial(128),  /* getrsx sub-functions */
        get$kill$pb byte initial(129),
        get$fcb$pb  byte initial(130),
        sstring(128) byte,  /* substitute string */
        sbp byte,   /* source buffer pointer */
        ssbp byte,  /* sub string buffer pointer */
        ver address,
        a address,  /* calling program's stack pointer */
        prog$flag based a address;

33  1   declare scbpd structure
       (offset byte,
        set    byte,
        value  address);

34  1   declare parse$fn structure
       (buff$adr address,
        fcb$adr  address);

35  1   declare subpb structure
      (io$type      byte,
       echo$flag    byte,
       filtered$flag byte,
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
            program$flag  byte)
            initial (con$type,true,true,false);


 36  1      declare
                ctrlc           literally '3',
                ctrlx           literally '18h',
                bksp            literally '8',
                submit$file$drv literally '15';


            /****************************************
            *                                       *
            * B D O S   F U N C T I O N   C A L L S *
            *                                       *
            ****************************************/



 37  1      printchar:
                procedure(char);
 38  2          declare char byte;
 39  2          call mon1(2,char);
 40  2          end printchar;

 41  1      conin:
                procedure byte;
 42  2          return mon2(6,0fdh);
 43  2          end conin;

 44  1      print: procedure(a);
 45  2          declare a address;
                /* print the string starting at address a until the
                next dollar sign is encountered */
 46  2          call mon1(9,a);
 47  2          end print;

 48  1      read$console$buf: procedure (buffer$address,max) byte;
 49  2          declare buffer$address address;
 50  2          declare new$max based buffer$address address;
 51  2          declare max byte;
 52  2          new$max = max;
 53  2          call mon1(10,buffer$address);
 54  2          buffer$address = buffer$address + 1;
 55  2          return new$max; /* actually number of characters input */
 56  2      end read$console$buf;

 57  1      version: procedure address;
                /* returns current cp/m version */
 58  2          return mon3(12,0);
 59  2          end version;

 60  1      open: procedure(fcb) address;
 61  2          declare fcb address;
 62  2          return (mon3(15,fcb));
 63  2          end open;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
 64  1     close: procedure(fcb);
 65  2         declare fcb address;
 66  2         dcnt = mon2(16,fcb);
 67  2         end close;

 68  1     delete: procedure(fcb);
 69  2         declare fcb address;
 70  2         call mon1(19,fcb);
 71  2         end delete;

 72  1     diskread: procedure(fcb) byte;
 73  2         declare fcb address;
 74  2         return mon2(20,fcb);
 75  2         end diskread;

 76  1     diskwrite: procedure(fcb) byte;
 77  2         declare fcb address;
 78  2         return mon2(21,fcb);
 79  2         end diskwrite;


 80  1     ranread: procedure(fcb) byte;
 81  2         declare fcb address;
 82  2         return mon2(33,fcb);
 83  2         end ranread;

 84  1     make: procedure(fcb);
 85  2         declare fcb address;
 86  2         dcnt = mon2(22,fcb);
 87  2         end make;

 88  1     setdma: procedure(dma);
 89  2       declare dma address;
 90  2       call mon1(26,dma);
 91  2       end setdma;

 92  1     errormode: procedure(mode);
 93  2       declare mode byte;
 94  2       call mon2a(45,mode);
 95  2       end errormode;

 96  1     getscbbyte: procedure (offset) byte;
 97  2         declare offset byte;
 98  2         scbpd.offset = offset;
 99  2         scbpd.set = 0;
100  2         return mon2(49,.scbpd);
101  2     end getscbbyte;

102  1     setscbbyte:
               procedure (offset,value);
103  2         declare offset byte;
104  2         declare value byte;
105  2         scbpd.offset = offset;
106  2         scbpd.set = 0ffh;
107  2         scbpd.value = double(value);
108  2         call mon1(49,.scbpd);
109  2      end setscbbyte;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
110  1     rsx$call: procedure (rsxpb) address;
           /* call Resident System Extension */
111  2       declare rsxpb address;
112  2       return mon3(60,rsxpb);
113  2     end rsx$call;

           /************************************************
           *                                              *
           *       M A I N   S U B R O U T I N E S        *
           *                                              *
           ************************************************/

114  1     move: procedure(s,d,n);
115  2         declare (s,d) address, n byte;
116  2         declare a based s byte, b based d byte;
117  2             do while (n := n - 1) <> 255;
118  3             b = a; s = s + 1; d = d + 1;
121  3             end;
122  2         end move;
           /* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - */

123  1     crlf:   proc;
124  2         call printchar(cr);
125  2         call printchar(lf);
126  2         end crlf;
           /* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - */

127  1     bad$file:  proc;
128  2        call print(.('Invalid file name $'));
129  2        call mon1(0,0);
130  2     end bad$file;
           /* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - */


           /* fill string @ s for c bytes with f */
131  1     fill: procedure(s,f,c);
132  2       declare s address;
133  2       declare (f,c) byte;
134  2       declare a based s byte;
135  2       do while (c:=c-1) <> 255;
136  3         a=f;
137  3         s=s+1;
138  3       end;
139  2     end fill;
           /* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - */

140  1     error: procedure(a);
141  2         declare a address;
142  2         call crlf;
143  2         call print(.('Error On Line $'));
144  2         call print(.lnl);
145  2         call print(a);
146  2         call move(.dfcb(0),.sfcb(0),33);
147  2         call delete(.sfcb(0));  /* cleanup before exit */
148  2         call mon1(0,0);
               /* return to ccp */
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
149  2        end error;
             /* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - */

150  1      ucase: procedure (char) byte;
151  2        declare char byte;
152  2        if char >= 'a' then
153  2          if char < '{' then
154  2            return (char-20h);
155  2        return char;
156  2      end ucase;

             /* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - */

157  1      getucase: procedure byte;
158  2        declare c byte;
159  2        c = ucase(conin);
160  2        return c;
161  2      end getucase;
             /* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - */

162  1      getpasswd: procedure;
163  2        declare (i,c) byte;
164  2        call crlf;
165  2        call crlf;
166  2        call print(.('Enter Password: $'));
167  2      retry:
           call fill(.fcb16,' ',8);
168  2        do i=0 to 7;
169  3      nxtchr:
           if (c:=getucase) >= ' ' then
170  3          fcb16(i)=c;
171  3        if c = cr then
172  3          return;
173  3        if c = ctrlx then
174  3          go to retry;
175  3        if c = bksp then do;
177  4          if i < 1 then
178  4            goto retry;
179  4          else do;
180  5            fcb16(i := i - 1) = ' ';
181  5            goto nxtchr;
182  5            end;
183  4          end;
184  3        if c = 3 then
185  3          call mon1(0,0);
186  3        end;
187  2      end getpasswd;
             /* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - */

188  1      deblankparm: procedure;
             /* clear to next non-blank substitute string */
189  2        do while (sstring(ssbp) = ' ' or sstring(ssbp) = ctrli);
190  3          ssbp = ssbp + 1;
191  3        end;
192  2      end deblankparm;
             /* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - */
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
193  1     try$open: procedure;
194  2       declare error$code address;
195  2       call fill(.fcb16,' ',8);  /* blank storage for password */
196  2       if len0 <> 0 then
197  2           call move(pass0,.fcb16,len0);
198  2       call error$mode(0feh);
199  2       call setdma(.fcb16);  /* set dma to password */
200  2       error$code = open(.sfcb);
201  2       if low(error$code) = 0ffh then
202  2         if high(error$code) = 7 then do;
204  3           call getpasswd;
205  3           call crlf;
206  3           call setdma(.fcb16);
207  3           call error$mode(0);
208  3           error$code=open(.sfcb);
209  3         end;
210  2         else do;
211  3           if high(error$code) = 0 then
212  3             call print(.('ERROR: No ''SUB'' File Found$'));
213  3           call mon1(0,0);
214  3         end;
215  2       call setdma(.buff(0));
216  2       call error$mode(0);
217  2     end try$open;
           /* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - */

218  1     setup: procedure;
219  2         declare no$chars byte;
220  2         declare pstatus address;
221  2         declare b byte;
           /* move buffer to substitute string */
222  2         call move(.buff(1),.sstring(0),127);
223  2         sstring(buff(0))=0; /* mark end of string */
           /* check to see if there are parameters */
224  2         ssbp = 0;
225  2         call deblankparm;   /* skip over leading spaces */
226  2         if sstring(ssbp) = 0 then do;   /* no sub file, prompt for it */
228  3            call print(.('CP/M 3 SUBMIT Version 3.0',cr,lf,'$'));
229  3            call print(.('Enter File to SUBMIT: $'));
230  3            no$chars = read$console$buf(.buff(0),40);
231  3            buff(no$chars+2)=0;  /* mark end of input */
232  3            call crlf;
233  3            parse$fn.buff$adr = .buff(2);
234  3            parse$fn.fcb$adr = .sfcb(0);
235  3            pstatus = parse(.parse$fn);
236  3            if pstatus = 0FFFFh then
237  3               call bad$file;
238  3            call move(.buff(2),.sstring(0),127);
239  3         end;
240  2         call move(.('SUB'),.sfcb(9),3); /* set file type to SUB */
241  2         if sfcb(0) = 0 then
242  2            if (b:=getscbbyte(submit$file$drv)) > 0 then do;
244  3               sfcb(0)=b;      /* set file drive to that saved by CCP */
245  3               call setscbbyte(submit$file$drv,0);
246  3               end;
247  2         call try$open;
248  2         do while (sstring(ssbp) <> ' ' and sstring(ssbp) <> 0
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
                and sstring(ssbp) <> ctrli);
249   3           ssbp = ssbp + 1; /* skip over file name */
250   3         end;
251   2         call deblankparm;   /* skip over any spaces */
252   2         b = sstring(ssbp);
                /* File is open if this point reached */
253   2         sbp = 128; /* causes read below */

254   2         end setup;

255   1       getsource: procedure byte;
                /* read the next source character */
256   2         declare b byte;
257   2         if sbp > 127 then
258   2             do; if diskread(.sfcb(0)) <> 0 then
260   3                 return endfile;
261   3             sbp = 0;
262   3             end;
263   2         if (b := buff((sbp:=sbp+1)-1)) = cr then do;
                  /* increment line */
265   3           if (ln5:=ln5+1) > '9' then do;
267   4             ln5 = '0';
268   4             if (ln4:=ln4+1) > '9' then do;
270   5               ln4 = '0';
271   5               if (ln3:=ln3+1) > '9' then do;
273   6                 ln3 = '0';
274   6                 if (ln2:=ln2+1) > '9' then do;
276   7                   ln2 = '0';
277   7                   ln1 = ln1 + 1;
278   7                 end;
279   6               end;
280   5             end;
281   4           end;
282   3         end;
283   2         return b;
284   2         end getsource;

285   1       writebuff: procedure;
                /* write the contents of the buffer to disk */
286   2         if diskwrite(.dfcb) <> 0 then /* error */
287   2             call error(.('Disk Write Error$'));
288   2         end writebuff;

289   1       declare rbuff(2048) byte, /* jcl buffer */
                    rbp address,      /* jcl buffer pointer */
                    rlen byte;     /* length of current command */

290   .       fillrbuff: procedure;
291   2         declare s byte;  /* sub string buffer pointer */

292   2         notend: procedure byte;
                    /* look at next character in sstring, return
                    true if not at the end of the string - char passed
                    back in 's' */
293   3             if not ((s := sstring(ssbp)) = ' ' or s = 0) then
294   3                 do;
295   4                 ssbp = ssbp + 1;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
296  4              return true;
297  4              end;
298  3            return false;
299  3            end notend;

300  2         write$rbuff: procedure;
301  3           declare j byte;
302  3           declare i address;
303  3           rbp=0; i=0;
305  3           do while (i < 2048);
306  4             do j=0 to 127;
307  5               if rbuff(i+j)=sysin$endfile
                      then goto close$file;
309  5             end;
310  4             call setdma(.rbuff(i));
311  4             call writebuff;
312  4             i=i+128;
313  4           end;
314  3           call setdma(.buff(0));
315  3           return;

316  3         close$file:
              call setdma(.rbuff(i));
317  3           call writebuff;
318  3           call setdma(.buff(0));
319  3           drrec, drr2 = 0;          /* set to 1st record in file */
320  3           dcnt = ranread(.dfcb);        /* read to position at start */
321  3           if dcnt <> 0 then
322  3              call error(.('Random Read $'));
323  3           goto exit$from$process;
324  3         end write$rbuff;

325  2       putrbuff: procedure(b);
326  3            declare b byte;
327  3            if (rbp > last(rbuff)) then do;
329  4              call print(.(',$'));
330  4              call write$rbuff;
331  4            end;
332  3            rbuff(rbp) = b;
333  3            if b = sysin$endfile then
334  3              call write$rbuff;
335  3            rbp = rbp + 1;
336  3         end putrbuff;

337  2         declare (reading,b,newline,progline) byte;
          /* fill the jcl buffer */
338  2         rbp = 0;
339  2         reading = true;
340  2            do while reading;
341  3            rlen = 0; /* reset command length */
342  3            newline,progline = true;
343  3               do while (b:=getsource) <> endfile and b <> cr;
344  4               if b <> lf then
345  4                  do; if b = sysin$endfile then
347  5                        call error(.('Invalid ASCII Character$'));
348  5                     if newline then do; /* program input begins with < */
350  6                       newline = false;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
351  6                        if b <> '<' then
352  6                          progline = false;
353  6                       end;
354  5                       if b = '$' then /* copy substitute string */
355  5                         do; if (b:=getsource) = '$' then
                                 /* $$ replaced by $ */
357  6                            call putrbuff(b); else
358  6                         if (b := b - '0') > 9 then
359  6                            call error(.('Parameter Error$')); else
360  6                            do; /* find string 'b' in sstring */
361  7                            ssbp = 0; call deblankparm; /* ready to scan string */
363  7                               do while b <> 0; b = b - 1;
                                    /* clear next parameter */
365  8                                    do while notend;
366  9                                    end;
367  8                                 call deblankparm;
368  8                                 end;
                                  /* ready to copy substitute string from position ssbp */
369  7                                 do while notend;
370  8                                 call putrbuff(s);
371  8                                 end;
372  7                               end;
373  6                        end; else /* not a '$' */
374  5                      if b = '^' then do; /* possible control character */
376  6                        b=getsource;
377  6                        if b = '^' then
378  6                          call putrbuff('^'); /* '^^' ==> '^' */
379  6                        else do;
380  7                          if b < '@' then /* number symbols */
381  7                            call putrbuff(b-' ');
                               else
382  7                            if b < '`' then   /* upper case */
383  7                              call putrbuff(b-'@');
                                 else
384  7                              call putrbuff(b-'`');   /* lower case */
385  7                        end;
386  6                      end;
                        /* check for multiple commands <com>!<com>!<com> */
387  5                      else if b = '!' and not progline then do;
389  6                        call putrbuff(cr); /* mark eoln with cr, lf */
390  6                        call putrbuff(lf);
391  6                      end;
                        else /* not $ or ^ */
392  5                        call putrbuff(b);
393  5                    end;
394  4                  end; /* of line or input file - compute length */
395  3              reading = b = cr;
396  3              call putrbuff(cr); /* mark eoln with cr, lf */
397  3              call putrbuff(lf);
398  3              end;
                /* entire file has been read and processed */
399  2           rbp = rbp - 2;  /* back up; too many cr,lf's on last line */
400  2           call putrbuff(sysin$endfile);   /* mark end of file */
401  2           end fillrbuff;

402  1       makefile: procedure;
403  2           declare i byte;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
404  2          declare rsxadr addr;
405  2          declare rsxbase based rsxadr addr;

406  2          rsxadr = rsx$call(.get$init$pb);
407  2          i = high(rsxbase);      /* rsxbase = addr of kill flag */
408  2          i = shr(i,2);
409  2          dfcb(6) = i/10 + '0';
410  2          dfcb(7) = i mod 10 + '0';
411  2          call errormode(0ffh);   /* set to return errors */
412  2          drec = 0; /* zero the next record to write */
413  2          call make(.dfcb);
414  2          if dcnt = 255 then do;
416  3            call delete(.dfcb);  /* file might exist */
417  3            call errormode(0);
418  3            call make(.dfcb);    /* try make again */
419  3            if dcnt = 255 then do;
421  4              call print(.('ERROR: Directory Full$'));
422  4              call mon1(0,0);
423  4              end;
424  3            end;
425  2          call errormode(0);
426  2      end makefile;

           /***********************************************
           *                                              *
           *          M A I N   P R O G R A M             *
           *                                              *
           ***********************************************/

427  1      plm:
             ver = version;
428  1          if (low(ver) < cpmversion) or (high(ver) = 1) then do;
430  2            call print(.('Requires CP/M 3.0 $'));
431  2            call mon1(0,0);
432  2          end;
433  1          dfcb(0)=getscbbyte(temp$file$drive$offset);
434  1          call setup;
435  1          call makefile;
436  1          call fillrbuff;
437  1      exit$from$process:
             /* check if GET is above us and about to abort */
             a = rsx$call(.get$fcb$pb);
438  1          if a <> 0ffh then do;
440  2            a = a - 2;
441  2            if prog$flag then
442  2              a = rsx$call(.get$kill$pb);
443  2          end;
444  1          call move(.dfcb(0),.sfcb(0),33);   /* move to fcb @ 5ch */
445  1          call getf(.subpb);     /* GETF also does submit processing */
446  1      end sub;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

CROSS-REFERENCE LISTING

DEFN	ADDR	SIZE	NAME, ATTRIBUTES, AND REFERENCES	
44	00C4H	2	A	ADDRESS PARAMETER 45 46
116	0000H	1	A	BYTE BASED(S) 118
12	0000H	2	A	ADDRESS PARAMETER 13
15	0000H	1	A	BYTE PARAMETER 16
140	00E8H	2	A	ADDRESS PARAMETER 141 145
134	0000H	1	A	BYTE BASED(S) 136
32	0085H	2	A	ADDRESS 32 437 438 440 442
18	0000H	2	A	ADDRESS PARAMETER 19
31			ADDR	LITERALLY 404 405
256	00F4H	1	B	BYTE 263 283
116	0000H	1	B	BYTE BASED(D) 118
325	08FCH	1	B	BYTE PARAMETER 326 332 333
337	08FEH	1	B	BYTE 343 344 346 351 354 356 357 358 363 364 374 376 377 380 381 382 383 384 387 392 395
221	00F3H	1	B	BYTE 242 244 252
127	02F0H	15	BADFILE	PROCEDURE STACK=0004H 237
36			BKSP	LITERALLY 175

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
  7 0000H   128  BUFF              BYTE ARRAY(128) EXTERNAL(1)
                                    215  222  223  230  231  233  238  263  314  318

 34 0000H     2  BUFFADR           ADDRESS MEMBER(PARSEFN)
                                    233

 48 00C6H     2  BUFFERADDRESS     ADDRESS PARAMETER
                                     49   50   53   54

163 00EDH     1  C                 BYTE
                                    169  170  171  173  175  184

158 00EBH     1  C                 BYTE
                                    159  160

131 00E7H     1  C                 BYTE PARAMETER
                                    133  135

150 00EAH     1  CHAR              BYTE PARAMETER
                                    151  152  153  154  155

 37 00C3H     1  CHAR              BYTE PARAMETER
                                     38   39

 64 01E0H    19  CLOSE             PROCEDURE STACK=0002H

316 085AH        CLOSEFILE         LABEL
                                    308

  4 0000H     1  CMDRV             BYTE EXTERNAL(2)

 41 0187H     9  CONIN             PROCEDURE BYTE STACK=0002H
                                    159

 31               CONTYPE           LITERALLY
                                     35

 31               CPMVERSION        LITERALLY
                                    428

 31               CR                LITERALLY
                                    124  171  228  263  343  389  395  396

123 02E5H    11  CRLF              PROCEDURE STACK=0004H
                                    142  164  165  205  232

 31               CTLL              LITERALLY

 36               CTRLC             LITERALLY

 31               CTRLI             LITERALLY
                                    189  248

 36               CTRLX             LITERALLY
                                    173

114 00E1H     2  D                 ADDRESS PARAMETER
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
                                        115  116  120

    31              DCL                 LITERALLY

    32 002DH     1  DCNT                BYTE
                                          66   86  320  321  414  419

   188 041DH    37  DEBLANKPARM         PROCEDURE STACK=0002H
                                         225  251  362  367

    68 01F3H    16  DELETE              PROCEDURE STACK=0002H
                                         147  416

    32 0009H    36  DFCB                BYTE ARRAY(36) INITIAL
                                          32  146  286  320  409  410  413  416  418  433
                                         444

    72 0203H    16  DISKREAD            PROCEDURE BYTE STACK=0002H
                                         259

    76 0213H    16  DISKWRITE           PROCEDURE BYTE STACK=0002H
                                         286

    88 00D7H     2  DMA                 ADDRESS PARAMETER
                                          89   90

                    DOUBLE              BUILTIN
                                         107

    32 0029H     1  DREC                BYTE AT
                                         412

    32 002CH     1  DRR2                BYTE AT
                                         319

    32 002AH     2  DRREC               ADDRESS AT
                                         319

    35 0001H     1  ECHOFLAG            BYTE MEMBER(SUBPB)

    31              ENDFILE             LITERALLY
                                         260  343

   140 032AH    56  ERROR               PROCEDURE STACK=0006H
                                         287  322  347  359

   194 00EEH     2  ERRORCODE           ADDRESS
                                         200  201  202  208  211

    92 0256H    14  ERRORMODE           PROCEDURE STACK=0002H
                                         198  207  216  411  417  425

   437 0135H        EXITFROMPROCESS     LABEL
                                         323

    15 0000H     1  F                   BYTE PARAMETER
                                          16
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
131 00E6H     1  F                BYTE PARAMETER
                                   133  136

 12 0000H     1  F                BYTE PARAMETER
                                   13

 18 0000H     1  F                BYTE PARAMETER
                                   19

 31              FALSE            LITERALLY
                                   35  298  350  352

 76 00D1H     2  FCB              ADDRESS PARAMETER
                                   77  78

 72 00CFH     2  FCB              ADDRESS PARAMETER
                                   73  74

 68 00CDH     2  FCB              ADDRESS PARAMETER
                                   69  70

 64 00CBH     2  FCB              ADDRESS PARAMETER
                                   65  66

 60 00C9H     2  FCB              ADDRESS PARAMETER
                                   61  62

  5 0000H     1  FCB              BYTE ARRAY(1) EXTERNAL(3)

 84 00D5H     2  FCB              ADDRESS PARAMETER
                                   85  86

 80 00D3H     2  FCB              ADDRESS PARAMETER
                                   81  82

  6 0000H     1  FCB16            BYTE ARRAY(1) EXTERNAL(4)
                                   167  170  180  195  197  199  206

 34 0002H     2  FCBADR           ADDRESS MEMBER(PARSEFN)
                                   234

131 02FFH    43  FILL             PROCEDURE STACK=0002H
                                   167  195

290 0653H   376  FILLRBUFF        PROCEDURE STACK=000EH
                                   436

 35 0002H     1  FILTEREDFLAG     BYTE MEMBER(SUBPB)

 31              FOREVER          LITERALLY

 21 0000H     1  FUNC             BYTE PARAMETER
                                   22

 28 0000H        GETF             PROCEDURE EXTERNAL(15) STACK=0000H
                                   445
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
 32 0030H    1  GETFCBPB          BYTE INITIAL
                                   437

 32 002EH    1  GETINITPB         BYTE INITIAL
                                   406

 32 002FH    1  GETKILLPB         BYTE INITIAL
                                   442

162 038EH  143  GETPASSWD         PROCEDURE STACK=0006H
                                   204

 96 0264H   24  GETSCBBYTE        PROCEDURE BYTE STACK=0002H
                                   242  433

255 05C4H  125  GETSOURCE         PROCEDURE BYTE STACK=0004H
                                   343  356  376

157 0380H   14  GETUCASE          PROCEDURE BYTE STACK=0004H
                                   169

                HIGH              BUILTIN
                                   202  211  407  428

403 0901H    1  I                 BYTE
                                   407  408  409  410

163 00ECH    1  I                 BYTE
                                   168  170  177  180

302 08FAH    2  I                 ADDRESS
                                   304  305  307  310  312  316

 21 0000H    2  INFO              ADDRESS PARAMETER
                                    23

 28 0000H    2  INPUTTYPE         ADDRESS PARAMETER
                                    29

 35 0000H    1  IOTYPE            BYTE MEMBER(SUBPB)

301 08F9H    1  J                 BYTE
                                   306  307

                LAST              BUILTIN
                                   327

 31             LCA               LITERALLY

 31             LCZ               LITERALLY

  8 0000H    1  LEN0              BYTE EXTERNAL(6)
                                   196  197

 10 0000H    1  LEN1              BYTE EXTERNAL(8)

 31             LF                LITERALLY
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
                                  125  228  344  390  397

31                  LIT           LITERALLY
                                   31

32 0000H     9      LN            BYTE ARRAY(9) INITIAL
                                   32

32 0000H     1      LN1           BYTE AT
                                   144  277

32 0001H     1      LN2           BYTE AT
                                   274  276

32 0002H     1      LN3           BYTE AT
                                   271  273

32 0003H     1      LN4           BYTE AT
                                   268  270

32 0004H     1      LN5           BYTE AT
                                   265  267

                    LOW           BUILTIN
                                   201  428

84 0233H    19      MAKE          PROCEDURE STACK=0002H
                                   413  418

402 08CBH   129     MAKEFILE      PROCEDURE STACK=0004H
                                   435

48 00C8H     1      MAX           BYTE PARAMETER
                                   51   52

   0000H            MEMORY        BYTE ARRAY(0)

92 00D9H     1      MODE          BYTE PARAMETER
                                   93   94

12 0000H            MON1          PROCEDURE EXTERNAL(10) STACK=0000H
                                   39   46   53   70   90  108  129  148  185  213
                                  422  431

18 0000H            MON2          PROCEDURE BYTE EXTERNAL(12) STACK=0000H
                                   42   66   74   78   82   86  100

15 0000H            MON2A         PROCEDURE EXTERNAL(11) STACK=0000H
                                   94

21 0000H            MON3          PROCEDURE ADDRESS EXTERNAL(13) STACK=0000H
                                   58   62  112

114 02AEH   55      MOVE          PROCEDURE STACK=0002H
                                  146  197  222  238  240  444

114 00E3H    1      N             BYTE PARAMETER
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
                                115  117

337 08FFH     1  NEWLINE         BYTE
                                342  348  350

 50 0000H     2  NEWMAX          ADDRESS BASED(BUFFERADDRESS)
                                 52   55

219 00F0H     1  NOCHARS         BYTE
                                230  231

292 07CBH    44  NOTEND          PROCEDURE BYTE STACK=0002H
                                365  369

169 03B3H        NXTCHR          LABEL
                                181

 33 0000H     1  OFFSET          BYTE MEMBER(SCBPD)
                                 98  105

102 00D8H     1  OFFSET          BYTE PARAMETER
                                103  105

 96 00DAH     1  OFFSET          BYTE PARAMETER
                                 97   98

 60 01D0H    16  OPEN            PROCEDURE ADDRESS STACK=0002H
                                200  208

 25 0000H        PARSE           PROCEDURE ADDRESS EXTERNAL(14) STACK=0000H
                                235

 34 00BBH     4  PARSEFN         STRUCTURE
                                233  234  235

  7 0000H     2  PASS0           ADDRESS EXTERNAL(5)
                                197

  9 0000H     2  PASS1           ADDRESS EXTERNAL(7)

 25 0000H     2  PFCB            ADDRESS PARAMETER
                                 26

  2 00F8H        PLM             LABEL PUBLIC
                                427

 44 0190H    16  PRINT           PROCEDURE STACK=0002H
                                128  143  144  145  166  212  228  229  329  421
                                430

 37 0177H    16  PRINTCHAR       PROCEDURE STACK=0002H
                                124  125

 31              PROC            LITERALLY
                                123  127

 32 0000H     2  PROGFLAG        ADDRESS BASED(A)
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
                                    441

337 0900H     1  PROGLINE            BYTE
                                     342  352  387

 35 0003H     1  PROGRAMFLAG         BYTE MEMBER(SUBPB)

220 00F1H     2  PSTATUS             ADDRESS
                                     235  236

325 0894H    55  PUTRBUFF            PROCEDURE STACK=000CH
                                     357  370  378  381  383  384  389  390  392  396
                                     397  400

 80 0223H    16  RANREAD             PROCEDURE BYTE STACK=0002H
                                     320

289 08F5H     2  RBP                 ADDRESS
                                     303  327  332  335  338  399

289 00F5H  2048  RBUFF               BYTE ARRAY(2048)
                                     307  310  316  327  332

 48 01A0H    39  READCONSOLEBUF      PROCEDURE BYTE STACK=0002H
                                     230

337 08FDH     1  READING             BYTE
                                     339  340  395

167 039AH        RETRY               LABEL
                                     174  178

289 08F7H     1  RLEN                BYTE
                                     341

404 0902H     2  RSXADR              ADDRESS
                                     405  406

405 0000H     2  RSXBASE             ADDRESS BASED(RSXADR)
                                     407

110 029EH    16  RSXCALL             PROCEDURE ADDRESS STACK=0002H
                                     406  437  442

110 00DDH     2  RSXPB               ADDRESS PARAMETER
                                     111  112

291 08F8H     1  S                   BYTE
                                     293  370

131 00E4H     2  S                   ADDRESS PARAMETER
                                     132  134  137

114 00BFH     2  S                   ADDRESS PARAMETER
                                     115  116  119

 32 00B1H     1  SBP                 BYTE
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
                                    253  257  261  263

 33 00B7H     4  SCBPD              STRUCTURE
                                     98   99  100  105  106  107  108

 33 0001H     1  SET                BYTE MEMBER(SCBPD)
                                     99  106

 88 0246H    16  SETDMA             PROCEDURE STACK=0002H
                                    199  206  215  310  314  316  318

102 027CH    34  SETSCBBYTE         PROCEDURE STACK=0002H
                                    245

218 04C9H   251  SETUP              PROCEDURE STACK=000AH
                                    434

  3 0000H    33  SFCB               BYTE ARRAY(33) EXTERNAL(0)
                                    146  147  200  208  234  240  241  244  259  444

                 SHR                BUILTIN
                                    408

 32 00B2H     1  SSBP               BYTE
                                    189  190  224  226  248  249  252  293  295  361

 32 0031H   128  SSTRING            BYTE ARRAY(128)
                                    189  222  223  226  238  248  252  293

  1 00F5H   130  SUB                PROCEDURE STACK=0010H

 36              SUBMITFILEDRV      LITERALLY
                                    242  245

 35 00BFH     4  SUBPB              STRUCTURE INITIAL
                                    445

 31              SYSINENDFILE       LITERALLY
                                    307  333  346  400

 11 0000H     1  TBUFF              BYTE ARRAY(1) EXTERNAL(9)

 31              TEMPFILEDRIVEOFFSET LITERALLY
                                    433

 31              TRUE               LITERALLY
                                     35  296  339  342

193 0442H   135  TRYOPEN            PROCEDURE STACK=0008H
                                    247

150 0362H    30  UCASE              PROCEDURE BYTE STACK=0000H
                                    159

 33 0002H     2  VALUE              ADDRESS MEMBER(SCBPD)
                                    107
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
  102 00DCH     1  VALUE              BYTE PARAMETER
                                      104  107

   32 00B3H     2  VER                ADDRESS
                                      427  428

   57 01C7H     9  VERSION            PROCEDURE ADDRESS STACK=0002H
                                      427

   31              WHAT               LITERALLY

  285 0641H    18  WRITEBUFF          PROCEDURE STACK=0008H
                                      311  317

  300 07F7H   157  WRITERBUFF         PROCEDURE STACK=000AH
                                      330  334
```

```
MODULE INFORMATION:

     CODE AREA SIZE     = 094CH   2380D
     VARIABLE AREA SIZE = 0904H   2308D
     MAXIMUM STACK SIZE = 0010H     16D
     671 LINES READ
     0 PROGRAM ERROR(S)

END OF PL/M-80 COMPILATION
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
0000 SUBMIT
0000 MCD80A
0004 BDISK    0080 BUFF     0080 BUFFA    0050 CMDRV    0000 CPU
007C CR             006D DOLLA     005C FCB       006C FCB16     005C FCBA
005C IFCB     005C IFCBA    0003 IOBYTE   0053 LEN0     0056 LEN1
0006 MAXB     0006 MEMSIZ   0005 MON1     0005 MON2     0005 MON2A
0005 MON3     0000 OFFSET   006E PARMA     0051 PASS0    0054 PASS1
007F R0             007D RR        007D RRECA     007F RRECO     005C SFCB
0080 TBUFF
0000 SUB
17E3 MEMORY   0278 PLM      0EDF LN       0EDF LN1      0EE0 LN2
0EE1 LN3      0EE2 LN4      0FE3 LN5      0EE8 DFCB     0F08 DREC
0F09 DRREC    0F0B DRR2     0F0C DCNT     0F0D GETINITPB 0F0E GETKILLPB
0F0F GETFCBPB 0F10 SSTRING  0F90 SBP      0F91 SSBP     0F92 VER
0F94 A        0F96 SCBPD    0F9A PARSEFN  0F9E SUBPB    02F7 PRINTCHAR
0FA2 CHAR     0307 CONIN    0310 PRINT    0FA3 A
0320 READCONSOLEBUF         0FA5 BUFFERADDRESS          0FA7 MAX
0347 VERSION  0350 OPEN     0FA8 FCB      0360 CLOSE    0FAA FCB
0373 DELETE   0FAC FCB      0383 DISKREAD 0FAE FCB      0393 DISKWRITE
0FB0 FCB      03A3 RANREAD  0FB2 FCB      03B3 MAKE     0FB4 FCB
03C6 SETDMA   0FB6 DMA      03D6 ERRORMODE 0FB8 MODE    03E4 GETSCBBYTE
0FB9 OFFSET   03FC SETSCBBYTE 0FBA OFFSET 0FBB VALUE    041E RSXCALL
0FBC RSXPB    042E MOVE     0FBE S        0FC0 D        0FC2 N
0465 CRLF     0470 BADFILE  047F FILL     0FC3 S        0FC5 F
0FC6 C        04AA ERROR    0FC7 A        04E2 UCASE    0FC9 CHAR
0500 GETUCASE 0FCA C        050E GETPASSWD 0FCB I       0FCC C
051A RETRY    0533 NXTCHR   059D DEBLANKPARM            05C2 TRYOPEN
0FCD ERRORCODE 0649 SETUP   0FCF NOCHARS  0FD0 PSTATUS  0FD2 B
0744 GETSOURCE 0FD3 B       07C1 WRITEBUFF 0FD4 RBUFF   17D4 RBP
17D6 RLEN     07D3 FILLRBUFF 17D7 S       094B NOTEND   0977 WRITERBUFF
17D8 J        17D9 I        09DA CLOSEFILE 0A14 PUTRBUFF 17DB B
17DC READING  17DD B        17DE NEWLINE  17DF PROGLINE  0A4B MAKEFILE
17E0 I        17E1 RSXADR   02B5 EXITFROMPROCESS
0000 PARSE
0C15 DBLNK1   0C09 DEBLNK   0BC7 DELIN    0C02 DELIM1   0C05 DELIM2
0BBF PADFLD   0B5D PARS81   0B5E PARS82   0ACC PARSE    0AED PARSE1
0B1B PARSE2   0B1E PARSE3   0B30 PARSE4   0B34 PARSE5   0B36 PARSE6
0B4A PARSE7   0B4E PARSE8   0B6F PARSE9   0B98 PWFLD    0BAB PWFLD1
0BBA SETFD1   0B8C SETFD2   0B90 SETFD3   0B75 SETFLD
0000 GETF
FFFF BIOSFU   007F BIOSON   00B4 CCPFLG   0001 CINF     0010 CLOSEF
00F9 COMMON   0074 CONINF   00CF CONMOD   007A CONOUF   006E CONSTF
0002 COUTF    000D CR       0006 CRAWF    000A CREADF   000B CSTATF
0019 CURDRV   0013 DELF     001A DMAF     0014 DREADF   00AA ERRFLG
0000 EVERYT   0000 FALSE    0080 INITF    008D JKILLF   00C3 JMPI
0081 KILLF    000A LF       00D4 LISTCP   0080 LISTFX   0021 LXIH
00FF NOREST   000F OPENF    0009 PBUFF    0005 PCHRF    0098 REALDO
003C RSXF     003A SCBA     0031 SCBF     0022 SHLDI    0080 STFIX
FFFF TRUE     0020 USERF    0000 WBOOT    0001 WBOOTA   0068 WBOOTF
0D2B BIOSER   0DF8 BIOSIN   0DFA BIOSMO   0DF6 BIOSTA   0E00 CONINJ
0DFC CONSTJ   0E07 COOKED   0D6D DAT[illegible]9 0E06 ECHO   0D3F ERROR
0D44 ERROR1   0CBD FIX0     0CCA FIX1     0CDD FIX2     0CE7 FIX3
0C19 GETF     0D65 GETSCB   0D5D GETUSR   0DF4 INITTA   0D7A JOURNK
0D6F LDIR     0DD7 MEMERR   0E2B MOVEND   0C52 MOVFCB   0DF4 MOVSTA
0D7C NOBIOS   0D57 NOMEM    0DA1 NOTDON   0D3C NOTIMP   0E08 PROGRA
0E03 REALBD   0DFB RESTOR   0D7B RSXINI   0D15 RSXMOV   0DF4 SCBADR
0D5F SETUSR   0E2A SUBCR    0E0A SUBFCB   0E18 SUBMOD   0E19 SUBRC
0E09 SUBUSR   0CE9 TRAP
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #

```
0000 SUBMIT$
0000 MCD80A$
0000 SUB$
02F7 37          02FB 39          0306 40          0307 41          0307 42
0310 43          0310 44          0316 46          031F 47          0320 48
0328 52          0332 53          033B 54          0342 55          0347 56
0347 57          0347 58          0350 59          0350 60          0356 62
0360 63          0360 64          0366 66          0372 67          0373 68
0379 70          0382 71          0383 72          0389 74          0393 75
0393 76          0399 78          03A3 79          03A3 80          03A9 82
03B3 83          03B3 84          03B9 86          03C5 87          03C6 88
03CC 90          03D5 91          03D6 92          03DA 94          03E3 95
03E4 96          03E8 98          03EE 99          03F3 100         03FC 101
03FC 102    0402 105    0408 106    040D 107    0415 108
041D 109    041E 110    0424 112    042E 113    042E 114
043D 117    0449 118    0453 119    045A 120    0461 121
0464 122    0465 123    0465 124    046A 125    046F 126
0470 127    0470 128    0476 129    047E 130    047F 131
048C 135    0498 13     049F 137    04A6 138    04A9 139
04AA 140    04B0 142    04B3 143    04B9 144    04BF 145
04C7 146    04D3 147    04D9 148    04E1 149    04E2 150
04E6 152    04EE 153    04F6 154    04FC 155    0500 156
0500 157    0500 159    050A 160    050E 161    050E 162
050E 164    0511 165    0514 166    051A 167    0525 168
0533 169    053E 170    054B 171    0553 172    0554 173
055C 174    055F 175    0567 177    056F 178    0572 180
0582 181    0585 182    0585 183    0585 184    058D 185
0595 186    059C 187    059D 188    059D 189    05BA 190
05BE 191    05C1 192    05C2 193    05C2 195    05CD 196
05D5 197    05E3 198    05E8 199    05EE 200    05F7 201
0600 202    0609 204    060C 205    060F 206    0615 207
061A 208    0623 209    0626 211    062F 212    0635 213
063D 214    063D 215    0643 216    0648 217    0649 218
0649 222    0655 223    0660 224    0665 225    0668 226
0677 228    067D 229    0683 230    068E 231    0699 232
069C 233    06A2 234    06A8 235    06B1 236    06BE 237
06C1 238    06CD 239    06CD 240    06D9 241    06E1 242
06F0 244    06F6 245    06FD 246    06FD 247    0700 248
0727 249    072B 250    072E 251    0731 252    073E 253
0743 254    0744 255    0744 257    074D 259    0758 260
075B 261    0760 262    0760 263    0778 265    0786 267
078B 268    0796 270    079B 271    07A6 273    07AB 274
07B6 276    07BB 277    07BD 278    07BD 279    07BD 280
07BD 281    07BD 282    07BD 283    07C1 284    07C1 285
07C1 286    07CC 287    07D2 288    07D3 290    094B 292
094B 293    096D 295    0971 296    0974 297    0974 298
0977 299    0977 300    0977 303    097D 304    0980 305
098C 306    099A 307    09AD 308    09B0 309    09B7 310
09C3 311    09C6 312    09D0 313    09D3 314    09D9 315
09DA 316    09E6 317    09E9 318    09EF 319    09F9 320
0A02 321    0A0A 322    0A10 323    0A13 324    0A14 325
0A18 327    0A24 329    0A2A 330    0A2D 331    0A2D 332
0A38 333    0A40 334    0A43 335    0A4A 336    07D3 338
07D9 339    07DE 340    07E5 341    07EA 342    07F2 343
080D 344    0815 346    081D 347    0823 348    082A 350
082F 351    0836 352    083B 353    083B 354    0843 356
084E 357    0858 358    0867 359    0870 361    0875 362
0878 363    0880 364    0884 365    088B 366    088E 367
0891 368    0894 369    089B 370    08A2 371    08A5 372
08A5 373    08A8 374    08B0 376    08B6 377    08BE 378
08C6 380    08CE 381    08DA 382    08E2 383    08EE 384
08F7 385    08F7 386    08FA 387    090E 389    0913 390
0918 391    091B 392    0922 393    0922 394    0925 395
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER: #_____

```
0930 396    0935 397    093A 398    093D 399    0945 400
094A 401    0A4B 402    0A4B 406    0A54 407    0A5C 408
0A63 409    0A75 410    0A87 411    0A8C 412    0A91 413
0A97 414    0A9F 416    0AA5 417    0AAA 418    0AB0 419
0AB8 421    0ABE 422    0AC6 423    0AC6 424    0AC6 425
0ACB 426    0275 427    0281 428    0296 430    029C 431
02A4 432    02A4 433    02AC 434    02AF 435    02B2 436
02B5 437    02C1 438    02CD 440    02D5 441    02DA 442
02E3 443    02E3 444    02EF 445    02F5 446
0000 PARSE#

0000 GETF#
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____ ____

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____ ____

```
                title 'GET.RSX 3.0 - CP/M 3.0 Input Redirection - August 1982'
                ;*****************************************************************
                ;
                ;       get  'Input Redirection Facility'  version 3.0
                ;
                ;       11/30/82 - Doug Huskey
                ;       This RSX redirects console input and status from a file.
                ;*****************************************************************
                ;
                ;
FFFF =          true            equ     0ffffh
0000 =          false           equ     00000h
                ;
FFFF =          submit          equ     true    ;true if submit RSX
0000 =          remove$rsx      equ     false   ;true if RSX removes itself
                ;                               ;false if LOADER does removes
                ;
                ;
                ;       generation procedure
                ;
                ;       rmac getrsx
                ;       xref getrsx
                ;       link getrsx[op]
                ;       ERA get.RSX
                ;       REN get.RSX=getRSX.PRL
                ;       GENCOM $1.COM get.RSX           ($1 is either SUBMIT or GET)
                ;
                ;
                ;       initialization procedure
                ;
                ;       GETF makes a RSX function 60 call with a sub-function of
                ;       128.  GETRSX returns the address of a data table containing:
                ;
                ;       init$table:
                ;               dw      kill            ;RSX remove flag addr in GET
                ;               dw      bios$constat    ;bios entry point in GET
                ;               dw      bios$conin      ;bios entry point in GET
                ;
                ;       GETF initializes the data are between movstart: and movend:
                ;       and moves it into GET.RSX.  This means that data should not
                ;       be reordered without also changing GETF.ASM.
                ;
FFFF =          bios$functions equ      true    ;intercept BIOS console functions
                ;
                ;       low memory locations
                ;
0000 =          wboot   equ     0000h
0005 =          bdos    equ     0005h
0006 =          bdosl   equ     bdos+1
0080 =          buf     equ     0080h
                ;
                ;       equates for non graphic characters
                ;
0003 =          ctlc    equ     03h     ; control c
0005 =          ctle    equ     05h     ; physical eol
0008 =          ctlh    equ     08h     ; backspace
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
 57  0010 =        ctlp   equ     10h     ; prnt toggle
 58  0012 =        ctlr   equ     12h     ; repeat line
 59  0013 =        ctls   equ     13h     ; stop/start screen
 60  0015 =        ctlu   equ     15h     ; line delete
 61  0018 =        ctlx   equ     18h     ; =ctl-u
 62                       if submit
 63  00FF =        ctlz   equ     0ffh
 64                       else
 65                ctlz   equ     1ah     ; end of file
 66                       endif
 67  007F =        rubout equ     7fh     ; char delete
 68  0009 =        tab    equ     09h     ; tab char
 69  000D =        cr     equ     0dh     ; carriage return
 70  000A =        lf     equ     0ah     ; line feed
 71  005E =        ctl    equ     5eh     ; up arrow
 72                ;
 73                ;       BDOS function equates
 74                ;
 75  0001 =        cinf   equ     1       ;read character
 76  0002 =        coutf  equ     2       ;output character
 77  0006 =        crawf  equ     6       ;raw console I/O
 78  000A =        creadf equ     10      ;read buffer
 79  000B =        cstatf equ     11      ;status
 80  0005 =        pchrf  equ     5       ;print character
 81  0009 =        pbuff  equ     9       ;print buffer
 82  000F =        openf  equ     15      ;open file
 83  0010 =        closef equ     16      ;close file
 84  0013 =        delf   equ     19      ;delete file
 85  0014 =        dreadf equ     20      ;disk read
 86  001A =        dmaf   equ     26      ;set dma function
 87  0020 =        userf  equ     32      ;set/get user number
 88  0031 =        scbf   equ     49      ;set/get system control block word
 89  003B =        loadf  equ     59      ;loader function call
 90  003C =        rsxf   equ     60      ;RSX function call
 91  0080 =        ginitf equ     128     ;GET initialization sub-function no.
 92  0081 =        gkillf equ     129     ;GET delete sub-function no.
 93  0082 =        gfcbf  equ     130     ;GET file display sub-function no.
 94  0084 =        pinitf equ     132     ;PUT initialization sub-funct no.
 95  0085 =        pckillf        equ     133     ;PUT CON: delete sub-function no.
 96  0086 =        pcfcbf equ     134     ;return PUT CON: fcb address
 97  0089 =        plkillf        equ     137     ;PUT LST: delete sub-function no.
 98  008A =        plfcbf equ     138     ;return PUT LST:fcb address
 99  008C =        gsigf  equ     140     ;signal GET without [SYSTEM] option
100  008D =        jinitf equ     141     ;JOURNAL initialization sub-funct no.
101  008E =        jkillf equ     142     ;JOURNAL delete sub-function no.
102  008F =        jfcbf  equ     143     ;return JOURNAL fcb address
103                ;
104                ;       System Control Block definitions
105                ;
106  003A =        scba   equ     03ah    ;offset of scbadr from SCB base
107  00B3 =        ccpflg equ     0b3h    ;offset of ccpflags word from page boundary
108  0020 =        ccpres equ     020h    ;ccp resident flag = bit 5
109  00FE =        bdosoff equ    0feh    ;offset of BDOS address from page boundary
110  00AC =        errflg equ     0ach    ;offset of error flag from page boundary
111  00C8 =        pg$mode        equ     0c8h    ;offset of page mode byte from pag. bound.
112  00C9 =        pg$def equ     0c9h    ;offset of page mode default from pag. bound.
```

COPYRIGHT ©1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
00CF =          conmode     equ     0cfh    ;offset of console mode word from pag. bound.
00D4 =          listcp equ  0d4h    ;offset of ^P flag from page boundary
00D8 =          dmaad  equ  0d8h    ;offset of DMA address from pg bnd.
00E0 =          usrcode     equ     0e0h    ;offset of user number from pg bnd.
00E1 =          dcnt   equ  0e1h    ;offset of dcnt, searcha & searchl from pg bnd.
006E =          constfx     equ     06eh    ;offset of constat JMP from page boundary
0074 =          coninfx     equ     074h    ;offset of conin JMP from page boundary


                ;*****************************************************************
                ;               RSX HEADER
                ;*****************************************************************

0000 0000000000serial:      db      0,0,0,0,0,0

                trapjmp:
0006 C34600             jmp     trap            ;trap read buff and DMA functions
0009 C30000     next:   jmp     0               ;go to BDOS
000C 0500       prev:   dw      bdos
000E FF         kill:   db      0FFh            ;0FFh => remove RSX at wstart
000F 00         nbank:  db      0
0010 4745542020rname:   db      'GET     '      ;RSX name
0018 0000       space:  dw      0
001A 00         patch:  db      0

                ;*****************************************************************
                ;               START OF CODE
                ;*****************************************************************

                ;
                ;       ABORT ROUTINE
                ;
                getout:
                        ;
                if bios$functions
                        ;
                        ;restore bios jumps
001B 3AF302             lda     restore$mode            ;may be FF, 7f, 80 or 0
001E 3C                 inr     a
001F C8                 rz                              ; FF = no bios interception
0020 2AF102             lhld    biosin
0023 EB                 xchg
0024 2AEF02             lhld    biosta
0027 CDF402             call    restore$bios            ;restore BIOS constat & conin jmps
002A F8                 rm                              ; 7f = RESBDOS jmps not changed
002B 2AED02             lhld    scbadr
002E 2E6E               mvi     l,constfx
0030 36C3               mvi     m,jmp
0032 E8                 rpe                             ; 80 = conin jmp not changed
0033 2E74               mvi     l,coninfx
0035 36C3               mvi     m,jmp
                endif
0037 C9                 ret                             ; 0  = everything done
                ;
                ;       ARRIVE HERE ON EACH BIOS CONIN OR CONSTAT CALL
                ;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

```
                        ;
                        bios$constat:
                                ;
                        if bios$functions
                                ;
                                ;enter here from BIOS constat
0038 010B04             lxi     b,4*256+cstatf  ;b=offset in exit table
003B C34C00             jmp     bios$trap
                        endif
                        ;
                        bios$conin:
                                ;
                        if bios$functions
                                ;
                                ;enter here from BIOS conin
003E 010606             lxi     b,6*256+crawf   ;b=offset in exit table
0041 1EFD               mvi     e,0fdh
0043 C34C00             jmp     biostrap
                        endif
                        ;
                        ;       ARRIVE HERE AT EACH BDOS CALL
                        ;
                        trap:
                                ;
                                ;
0046 21B803             lxi     h,excess
0049 0600               mvi     b,0
004B 70                 mov     m,b
                        biostrap:
                                ;enter here on BIOS calls

004C E1                 pop     h               ;return address
004D E5                 push    h               ;back to stack
004E 3A0800             lda     trap+jmp+2      ;GET.RSX page address
0051 BC                 cmp     h               ;high byte of return address
0052 DA7E00             jc      exit            ;skip calls on bdos above here
0055 79                 mov     a,c             ;function number
                                ;
                                ;
0056 FE0B               cpi     cstatf          ;status
0058 CADD00             jz      intercept
005B FE06               cpi     crawf
005D CADD00             jz      intercept       ;raw I/O
0060 21EA02             lxi     h,statflg       ;zero conditional status flag
0063 3600               mvi     m,0
0065 FE01               cpi     cinf
0067 CADD00             jz      intercept       ;read character
006A FE0A               cpi     creadf
006C CADD00             jz      intercept       ;read buffer
006F FE3C               cpi     rsxf
0071 CA8A00             jz      rsxfunc         ;rsx function
0074 FE1A               cpi     dmaf
0076 C27E00             jnz     exit            ;skip if not setting DMA
0079 EB                 xchg
007A 22A703             shld    udma            ;save user's DMA address
007D EB                 xchg
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
                        ;
                        exit:
                              ;go to real BDOS

                        if not bios$functions
                              ;
                              jmp     next            ;go to next RSX or BDOS

                        else
007E 78                       mov     a,b             ;get type of call:
007F 21EB02                   lxi     h,exit$table    ;0=BDOS call, 4=BIOS CONIN, 6=BIOS CONSTAT
0082 CD3D02                   call    addhla
0085 46                       mov     b,m             ;low byte to b
0086 23                       inx     h
0087 66                       mov     h,m             ;high byte to h
0088 68                       mov     l,b             ;HL = .exit routine
0089 E9                       pchl                    ;gone to BDOS or BIOS
                        endif
                        ;
                        ;
                        rsxfunc:                      ;check for initialize or delete RSX functions
008A 1A                       ldax    d               ;get RSX sub-function number
008B 21ED02                   lxi     h,init$table    ;address of area initialized by COM file
008E FE80                     cpi     ginitf
0090 C8                       rz
0091 3A0E00                   lda     kill
0094 B7                       ora     a
0095 C27E00                   jnz     exit
0098 1A                       ldax    d
0099 FE82                     cpi     gfcbf
009B 210303                   lxi     h,subfcb
009E C8                       rz
                        cksig:
009F FE8C                     cpi     gsigf
00A1 C2AB00                   jnz     ckKill
00A4 21A903                   lxi     h,get$active
00A7 3E81                     mvi     a,gKillf
00A9 96                       sub     m               ;toggle get$active flag
00AA 77                       mov     m,a             ;gKillf->0    0->gKillf

                        ckKill:
00AB FE81                     cpi     gKillf          ;remove this instance of GET?
00AD C27E00                   jnz     exit            ;jump if not


                        restor:
00B0 3AA903                   lda     get$active
00B3 B7                       ora     a
00B4 C8                       rz
00B5 CD1B00                   call    getout          ;bios jump fixup

                        if submit
00B8 0E10                     mvi     c,closef
00BA CDD701                   call    subdos
00BD 0E13                     mvi     c,delf
00BF CDD701                   call    subdos          ;delete SYSIN??.$$$ if not
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
                        endif
00C2 210E00             lxi     h,kill
00C5 35                 dcr     m               ;set to 0ffh, so we are removed
00C6 EB                 xchg                    ; D = base of this RSX
00C7 2AED02             lhld    scbadr
00CA 2EB4               mvi     l,ccpflg+1      ;hl = .ccp flag 2 in SCB
00CC 7E                 mov     a,m
00CD E6BF               ani     0bfh
00CF 77                 mov     m,a             ;turn off redirection flag
                        ;we must remove this RSX if it is the lowest one
00D0 3A0600             lda     bdosl+1         ;location 6 high byte
00D3 BA                 cmp     d               ;Does location 6 point to us
00D4 C0                 RNZ                     ;return if not
                        if remove$rsx
                        xchg                    ;D = scb page
                        lhld    next+1
                        shld    bdosl
                        xchg                    ;H = scb page
                        mvi     l,bdosoff       ;HL = 'BDOS' address in SCB
                        mov     m,e             ;put next address into SCB
                        inx     h
                        mov     m,d
                        xchg
                        mvi     l,0ch           ;HL = .previous RSX field in next RSX
                        mvi     m,7
                        inx     h
                        mvi     m,0             ;put previous into previous
                        ret
                        else
                        ;       CP/M 3 loader does RSX removal if DE=0
00D5 0E3B               mvi     c,loadf
00D7 110000             lxi     d,0
00DA C30900             jmp     next            ;ask loader to remove me
                        endif

                        ;
                        ;
                        ;       INTERCEPT EACH BDOS CONSOLE INPUT FUNCTION CALL HERE
                        ;
                        ;       enter with funct in A, info in DE
                        ;
                        intercept:
                        ;
00DD 3A0E00             lda     kill
00E0 B7                 ora     a
00E1 C27E00             jnz     exit            ;skip if remove flag turned on
                        ;
                        ;switch stacks
00E4 210000             lxi     h,0
00E7 39                 dad     sp
00E8 22B903             shld    old$stack
00EB 314104             lxi     sp,stack
00EE C5                 push    b               ;save function #
00EF D5                 push    d               ;save info
                        ;check redirection mode
00F0 CD2302             call    getmode         ;returns with H=SCB page
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
00F3 FE02          cpi     2
00F5 CA7501        jz      skip            ;skip if no redirection flag on

                   if submit
                   ;
                   ;       SUBMIT PROCESSOR
                   ;
                           ;check if CCP is calling
00F8 2EC8          ckccp:  mvi     l,pg$mode
00FA 74                    mov     m,H             ;set to non-zero for no paging
00FB 2EB4                  mvi     l,ccpflg+1      ;CCP FLAG 2 in SCB
00FD 7E                    mov     a,m             ;ccp flag byte 2 to A
00FE F640                  ori     040h
0100 77                    mov     m,a             ;set redirection flag on
0101 E620                  ani     ccpres          ;zero flag set if not CCP calling
0103 3A2603                lda     ccp$line
0106 CA2F01                jz      not$ccp
                           ;yes, CCP is calling
0109 B7                    ora     a
010A C23301                jnz     redirect        ;we have a CCP line
                           ;CCP & not a CCP line
010D E5                    push    h
010E CD9601                call    coninf          ;throw away until next CCP line
0111 21B803                lxi     h,excess
0114 7E                    mov     a,m
0115 B7                    ora     a               ;is this the first time?
0116 36FF                  mvi     m,true
0118 11BB03                lxi     d,garbage
011B 0E09                  mvi     c,pbuff
011D CC0900                cz      next            ;print the warning if so
0120 E1                    pop     h
0121 3A0E00                lda     Kill
0124 B7                    ora     a
0125 CAF800                jz      ckccp           ;get next character (unless eof)
0128 7E                    mov     a,m
0129 E67F                  ani     7fh             ;turn off disk reset (CCP) flag
012B 77                    mov     m,a
012C C30000                jmp     wboot           ;skip if remove flag turned on
                   ;
                   not$ccp:
                           ;no, its not the CCP
012F B7                    ora     a
0130 C27501                jnz     skip            ;skip if no program line

                   else
                           lda     program
                           ora     a               ;program input only?
                           mvi     l,ccpflg+1      ;CCP FLAG 2 in SCB
                           mov     a,m             ;ccp flag byte 2 to A
                           jz      set$no$page     ;jump if [system] option
                           ;check if CCP is calling
                           ani     ccpres          ;zero flag set if not CCP calling
                           jz      redirect        ;jump if not the CCP
                           lxi     h,ccpcnt        ;decrement once for each
                           dcr     m               ;time CCP active
                           cm      restor          ;if 2nd CCP appearance
```

CP/M-PLUS V3.0
COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. # CP3-135

```
                        lxi     d,cksig+1
                        mvi     c,rsxf          ;terminate any GETs waiting for
                        call    next            ;us to finish
                        jmp     skip
                        ;
                set$no$page:
                        ori     40h             ;A=ccpflag2, HL=.ccpflag2
                        mov     m,a             ;set redirection flag on
                        mvi     l,pg$mode
                        mov     m,h             ;set to non-zero for no paging
                endif
                        ;
                        ;       REDIRECTION PROCESSOR
                        ;
                redirect:
                        ;break if control-C typed on console
0133 CD6301             call    break
0136 D1                 pop     d
0137 C1                 pop     b               ;recover function no. & info
0138 C5                 push    b               ;save function
0139 D5                 push    d               ;save info
013A 79                 mov     a,c             ;function no. to A
013B 218601             lxi     h,retmon        ;program return routine
013E E5                 push    h               ;push on stack
                        ;
                        ;
013F FE0A               cpi     creadf
0141 CA8E02             jz      func10          ;read buffer (returns to retmon)
0144 FE01               cpi     cinf
0146 CA4202             jz      func1           ;read character (returns to retmon)
0149 FE0B               cpi     cstatf
014B CA6402             jz      func11          ;status (returns to retmon)
                ;
                func6:
                        ;direct console i/o - read if 0ffh
                        ;returns to retmon
014E 7B                 mov     a,e
014F 3C                 inr     a
0150 CA9101             jz      dirinp          ;0ffh in E for status/input
0153 3C                 inr     a
0154 CA6402             jz      CONBRK          ;0feh in E for status
0157 21EA02             lxi     h,statflg
015A 3600               mvi     m,0
015C 3C                 inr     a
015D CA9601             jz      coninf          ;0fdh in E for input
                        ;
                        ;direct output function
                        ;
0160 C37F01             jmp     skip1
                        ;
                break:  ;
                        ;quit if ^C typed
0163 0E0B               mvi     c,cstatf
0165 CDFC02             call    real$bdos
0168 B7                 ora     a               ;was ^C typed?
0169 C8                 rz
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
                        pop     h               ;throw away return address
016B CDB000             call    restor          ;remove this RSX, if so
016E 0E06               mvi     c,crawf
0170 1EFF               mvi     e,0ffh
0172 CD0900             call    next            ;eat ^C if not nested
                        ;
                skip:   ;
                        ;reset ^C status mode
0175 CD2302             call    getmode         ;returns .conmode+1
0178 2B                 dcx     h               ;hl = .conmode in SCB
0179 7E                 mov     a,m
017A E6FE               ani     0feh            ;turn off control C status
017C 77                 mov     m,a
                        ;restore the BDOS call
017D D1                 pop     d               ;restore BDOS function no.
017E C1                 pop     b               ;restore BDOS parameter
                        ;restore the user's stack
017F 2AB903     skip1:  lhld    old$stack
0182 F9                 sphl
0183 C37E00             jmp     exit            ;goto BDOS

                ;
                retmon:
                        ;normal entry point, char in A
0186 FEFF               cpi     ctlz
0188 CA7501             jz      skip
018B 2AB903             lhld    old$stack
018E F9                 sphl
018F 6F                 mov     l,a
0190 C9                 ret                     ;to calling program


                ;***************************************************************
                ;               BIOS FUNCTIONS (REDIRECTION ROUTINES)
                ;***************************************************************
                ;
                ;       ;direct console input
                dirinp:
0191 CD6402             call    conbrk
0194 B7                 ora     a
0195 C8                 rz
                ;
                ;
                ;       get next character from file
                ;
                        ;
                coninf:
                getc:   ;return ^Z if end of file
0196 AF                 xra     a
0197 212403             lxi     h,cbufp         ;cbuf index
019A 34                 inr     m               ;next chr position
019B FCD501             cm      readf           ;read a new record
019E B7                 ora     a
019F 06FF               mvi     b,ctlz          ;EOF indicator
01A1 C2AE01             jnz     getc1           ;jump if end of file
01A4 3A2403             lda     cbufp
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
 505  01A7 212703          lxi     h,cbuf
 506  01AA CD3D02          call    addhla          ;HL = .char
 507                       ;one character look ahead
 508                       ;new char in B, current char in nextchr
 509  01AD 46              mov     b,m             ;new character in B
 510  01AE 78        getc1: mov     a,b
 511  01AF FEFF            cpi     ctlz
 512  01B1 C5              push    b
 513  01B2 CCB000          cz      restor
 514  01B5 C1              pop     b
 515  01B6 212503          lxi     h,nextchr
 516  01B9 7E              mov     a,m             ;current character
 517  01BA FE0D            cpi     cr
 518  01BC 70              mov     m,b             ;save next character
 519  01BD C0              rnz
 520  01BE 78              mov     a,b             ;A=character after CR
 521  01BF FE0A            cpi     lf              ;is it a line feed
 522  01C1 CC9601          cz      getc            ;eat line feeds after a CR
 523                                               ;this must return from above
 524                                               ;rnz because nextchr = lf
 525                       ;
 526                  if submit
 527                       ;
 528  01C4 78              mov     a,b             ;get nextchr
 529  01C5 D63C            sui     '<'             ;program line?
 530  01C7 322603          sta     ccp$line        ;zero if so
 531  01CA CC9601          cz      getc            ;eat '<' char
 532                                               ;this must return from above
 533                                               ;rnz because nextchr = <
 534                  endif
 535  01CD 3E0D            mvi     a,cr            ;get back the cr
 536  01CF C9              ret                     ;with character in a
 537                  ;
 538                  ;       set DMA address in DE
 539                  ;
 540  01D0 0E1A       setdma: mvi     c,dmaf
 541  01D2 C30900          jmp     next
 542                  ;
 543                  ;       read next record
 544                  ;
 545  01D5 0E14       readf:  mvi     c,dreadf        ;read next record of input to cbuf
 546  01D7 C5         subdos: push    b
 547  01D8 112703          lxi     d,cbuf
 548  01DB CDD001          call    setdma          ;set DMA to our buffer
 549  01DE 2AED02          lhld    scbadr
 550  01E1 11AA03          lxi     d,sav$area      ;10 byte save area
 551  01E4 C1              pop     b               ;C  = function no.
 552  01E5 E5              push    h               ;save for restore
 553  01E6 D5              push    d               ;save for restore
 554  01E7 CD3202          call    mov7            ;save hash info in save area
 555  01EA 2EE0            mvi     l,usrcode       ;HL = .dcnt in SCB
 556  01EC CD3202          call    mov7            ;save dcnt, searcha & l, user# &
 557  01EF 2B              dcx     h               ;multi-sector I/O count
 558  01F0 3601            mvi     m,1             ;set multi-sector count = 1
 559  01F2 110203          lxi     d,subusr        ;DE = .submit user #
 560  01F5 2EE0            mvi     l,usrcode       ;HL = .BDOS user number
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
 561  01F7 1A          ldax   d
 562  01F8 77          mov    m,a
 563  01F9 13          inx    d
 564  01FA CD0900      call   next          ;read next record
 565  01FD E1          pop    h             ;HL = .sav$area
 566  01FE D1          pop    d             ;DE = .scb
 567  01FF F5          push   psw           ;save A (non-zero if error)
 568  0200 CD3202      call   mov7          ;restore hash info
 569  0203 1EE0        mvi    e,usrcode     ;DE = .dcnt in scb
 570  0205 CD3202      call   mov7          ;restore dcnt search addr & len
 571  0208 2AA703      lhld   udma
 572  020B EB          xchg
 573  020C CDD001      call   setdma        ;restore DMA to program's buffer
 574  020F AF          xra    a
 575  0210 322403      sta    cbufp         ;reset buffer position to 0
 576  0213 F1          pop    psw
 577  0214 B7          ora    a
 578  0215 C9          ret                  ;zero flag set, if successful
 579                ;
 580                ;     reboot from ^C
 581                ;
 582                rebootx:
 583                     ;store 0fffeh in clp$errcode in SCB
 584  0216 2AED02      lhld   scbadr
 585  0219 2EAC        mvi    l,errflg
 586  021B 36FE        mvi    m,0feh
 587  021D 23          inx    h
 588  021E 36FF        mvi    m,0ffh
 589  0220 C30000      jmp    wboot
 590                ;
 591                ;
 592                ;     get input redirection mode to A
 593                ;     turn on ^C status mode for break
 594                ;     return .conmode+1 in HL
 595                ;     preserve registers BC and DE
 596                ;
 597                getmode:
 598  0223 2AED02      lhld   scbadr
 599  0226 2ECF        mvi    l,conmode
 600  0228 7E          mov    a,m
 601  0229 F601        ori    1             ;turn on ^C status
 602  022B 77          mov    m,a
 603  022C 23          inx    h
 604  022D 7E          mov    a,m
 605  022E E603        ani    3             ;mask off redirection bits
 606  0230 3D          dcr    a             ;255=false, 0=conditional, 1=true,
 607  0231 C9          ret                  ;  2=don't redirect input
 608                ;
 609                ;     move routine
 610                ;
 611  0232 0607      mov7:  mvi    b,7
 612                       ;                HL = source
 613                       ;                DE = destination
 614                       ;                B = count
 615  0234 7E        move:  mov    a,m
 616  0235 12              stax   d
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
0236 23              inx     h
0237 13              inx     d
0238 05              dcr     b
0239 C23402          jnz     move
023C C9              ret
                     ;
                     ;       add a to hl
                     ;
023D 85         addhla:      add     l
023E 6F              mov     l,a
023F D0              rnc
0240 24              inr     h
0241 C9              ret
                     ;
                     ;*****************************************************************
                     ;               BDOS CONSOLE INPUT ROUTINES
                     ;*****************************************************************

                     ;
                     ;       February 3, 1981
                     ;
                     ;
                     ;       console handlers

0196 =          conin: equ     coninf
                     ;
                     conech:
                          ;read character with echo
0242 CD9601CD4F      call conin! call echoc! rc      ;echo character?
                          ;character must be echoed before return
0249 F5CD8302F1      push psw! call conout! pop psw
024E C9              ret                             ;with character in A
                     ;
                     echoc:
                          ;are we in cooked or raw mode?
024F 2100033534      lxi h,cooked! dcr m! inr m! rz  ;return if raw
                          ;echo character if graphic
                          ;cr, lf, tab, or backspace
0255 FE0DC8          cpi cr! rz              ;carriage return?
0258 FE0AC8          cpi lf! rz              ;line feed?
025B FE09C8          cpi tab! rz             ;tab?
025E FE08C8          cpi ctlh! rz            ;backspace?
0261 FE20C9          cpi ' '! ret            ;carry set if not graphic
                     ;
                     conbrk:         ;STATUS - check for character ready
0264 21EA02          lxi h,statflg
0267 4636FF          mov b,m! mvi m,0ffh     ;set conditional status flag true
026A CD2302          call getmode            ;check input redirection status mode
026D FE01C8          cpi 1! rz               ;actual status mode => return true
0270 B7C8            ora a! rz               ;false status mode  => return false
                          ;conditional status mode => false unless prev func was status
0272 78C9            mov a,b! ret_           ; return false if statflg false
                                             ; return true if statflg true
                     ;
                     ;
                     ctlout:
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
                        ;send character in A with possible preceding up-arrow
0274 CD4F02             call echoc              ;cy if not graphic (or special case)
0277 D28302             jnc conout              ;skip if graphic, tab, cr, lf, or ctlh
                                ;send preceding up arrow
                                push psw! mvi a,ctl! call conout ;up arrow
027A F53E5ECD83
0280 F1F640                     pop psw! ori 40h        ;becomes graphic letter
                                ;(drop through to conout)
                ;
                ;
                !       send character in A to console
                ;
                conout:
0283 5F                 mov     e,a
0284 3AFF02             lda     echo
0287 B7                 ora     a
0288 C8                 rz
0289 0E02               mvi     c,coutf
028B C30900             jmp     next
                ;
                ;
                read:   ;read to buffer address (max length, current length, buffer)
028E EB                 xchg                                    ;buffer address to HL
028F 4E23E50600         mov c,m! inx h! push h! mvi b,0         ;save .(current length)
                        ;B = current buffer length,
                        ;C = maximum buffer length,
                        ;HL= next to fill - 1
                        readnx:
                                ;read next character, BC, HL active
0294 C5E5                       push b! push h                          ;blen, cmax, HL saved
                                readn0:
0296 CD9601                             call conin                      ;next char in A
0299 E1C1                               pop h! pop b                    ;reactivate counters
029B FEFFC2A702                         cpi ctlz! jnz noteof            ;end of file?
02A0 0504CAE002                         dcr b! inr b! jz readen         ;skip if buffer empty
02A5 3E0D                               mvi a,cr                        ;otherwise return
                                noteof:
02A7 FE0DCAE002                         cpi cr!   jz readen             ;end of line?
02AC FE0ACAE002                         cpi lf!   jz readen             ;also end of line
02B1 FE10C2C602                         cpi ctlp! jnz notp              ;skip if not ctlp
                                        ;list toggle - change parity
02B6 E5C5                               push h! push b                  ;save counters
02B8 2AED022ED4                         lhld scbadr! mvi l,listcp       ;hl =.listcp
02BD 3E0196                             mvi a,1! sub m                  ;True-listcp
02C0 77                                 mov m,a                         ;listcp = not listcp
02C1 C1E1C39402                         pop b! pop h! jmp readnx        ;for another char
                                notp:
                                        ;not a ctlp
                                        ;place into buffer
                                rdecho:
02C6 2377                               inx h! mov m,a          ;character filled to mem
02C8 04                                 inr b                   ;blen = blen + 1
                                rdech1:
                                        ;look for a random control character
02C9 C5E5                               push b! push h          ;active values saved
02CB CD7402                             call ctlout             ;may be up-arrow C
02CE E1C17E                             pop h! pop b! mov a,m   ;recall char
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
02D1 FE03                       cpi ctlc            ;set flags for reboot test
02D3 78                         mov a,b             ;move length to A
02D4 C2DC02                     jnz notc            ;skip if not a control c
02D7 FE01                       cpi 1               ;control C, must be length 1
02D9 CA1602                     jz rebootx          ;reboot if blen = 1
                                ;length not one, so skip reboot
                        notc:
                                ;not reboot, are we at end of buffer?
02DC B9DA9402                   cmp c! jc readnx     ;go for another if not
                        readen:
                                ;end of read operation, store blen
02E0 E170                       pop h! mov m,b       ;M(current len) = B
02E2 F5                         push psw            ;may be a ctl-z
02E3 3E0DCD8302                 mvi a,cr! call conout ;return carriage
02E8 F1                         pop psw             ;restore character
02E9 C9                         ret
                ;
0242 =          func1: equ      conech
                        ;return console character with echo
                ;
                ;func6:         see intercept routine at front of module
                ;
028E =          func10:         equ     read
                        ;read a buffered console line
                ;
0264 =          func11: equ     conbrk
                        ;check console status
                ;
                ;

                ;*****************************************************************
                ;               DATA AREA
                ;*****************************************************************

02EA 00         statflg:        db      0       ;non-zero if prev funct was status
                        ;
                        ;

                ;*****************************************************************
                ;       Following variables and entry points are used by GET.COM
                ;       Their order and contents must not be changed without also
                ;       changing GET.COM.
                ;*****************************************************************
                        ;
                        if bios$functions
                        ;
                exit$table:                     ;addresses to go to on exit
02EB 0900               dw      next            ;BDOS
                        endif
                        ;
                movstart:
                init$table:                     ;addresses used by GET.COM for
02ED 0E00       scbadr:         dw      Kill            ;address of System Control Block
                        ;
                        if bios$functions       ;GET.RSX initialization
                        ;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
02EF 3800      biosta dw      bios$constat    ;set to real BIOS routine
02F1 3E00      biosin dw      bios$conin      ;set to real BIOS routine
                      ;
                                             ;restore only if changed when removed.
               restore$mode
02F3 00               db      0               ;if non-zero change LXI @jmpadr to JMP
                                             ;when removed.
               restore$bios:
                      ;hl = real constat routine
                      ;de = real conin routine
02F4 220000           shld    0               ;address of const jmp initialized by COM
02F7 EB               xchg
02FB 220000           shld    0               ;address of conin jmp initialized by COM
02FB C9               ret
                      endif
                      ;
               real$bdos:
02FC C30500           jmp     bdos            ;address filled in by COM
                      ;
                      ;
02FF 01        echo:  db      1
0300 00        cooked:        db      0
                      ;
               program:
0301 00               db      0               ;true if program input only
0302 00        subusr:        db      0               ;user number for redirection file
0303 01        subfcb:        db      1               ;a:
0304 535953494E       db      'SYSIN   '
030C 535542           db      'SUB'
030F 0000             db      0,0
0311 00        submod:        db      0
0312           subrc: ds      1
0313                  ds      16              ;map
0323           subcr: ds      1
                      ;
               movend:
               ;*****************************************************************

0324 80        cbufp  db      128             ;current character position in cbuf
0325 0D        nextchr        db      cr              ;next character (1 char lookahead)

                      if submit
               ccp$line:
0326 00               db      false           ;nonzero if line is for CCP
                      endif

               cbuf:                          ;128 byte record buffer

0327 0303030303       db      3,3,3,3, 3,3,3,3, 3,3,3,3, 3,3,3,3
0337 0303030303       db      3,3,3,3, 3,3,3,3, 3,3,3,3, 3,3,3,3
0347 0303030303       db      3,3,3,3, 3,3,3,3, 3,3,3,3, 3,3,3,3
0357 0303030303       db      3,3,3,3, 3,3,3,3, 3,3,3,3, 3,3,3,3

0367 0303030303       db      3,3,3,3, 3,3,3,3, 3,3,3,3, 3,3,3,3
0377 0303030303       db      3,3,3,3, 3,3,3,3, 3,3,3,3, 3,3,3,3
0387 0303030303       db      3,3,3,3, 3,3,3,3, 3,3,3,3, 3,3,3,3
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #

```
 841  0397 0303030303        db      3,3,3,3, 3,3,3,3, 3,3,3,3,3
 842
 843  03A7 8000        udma:  dw      buf             ;user dma address
 844                  get$active:
 845  03A9 81                 db      gKillf
 846                          ;
 847                  sav$area:                       ;14 byte save area (searchn)
 848  03AA 6868686868         db      68h,68h,68h,68h,68h, 68h,68h,68h,68h,68h
 849  03B4 68686868           db      68h,68h,68h,68h
 850  03B8 00          excess:        db      0
 851                  old$stack:
 852  03B9 0000               dw      0
 853                          if      submit
 854                  garbage:
 855                  ;       db      cr,lf
 856  03BB 5741524E49         db      'WARNING: PROGRAM INPUT IGNORED',cr,lf,'$'
 857                          else
 858                  ccpcnt:         db      1
 859                          endif
 860                  patch$area:
 861  03DC                    ds      30h
 862  040C 2031353132         db      ' 151282 '
 863  0414 20434F5059         db      ' COPYR ''82 DRI '
 864  0423 6767676767         db      67h,67h,67h,67h,67h, 67h,67h,67h,67h,67h
 865  042D 6767676767         db      67h,67h,67h,67h,67h, 67h,67h,67h,67h,67h
 866  0437 6767676767         db      67h,67h,67h,67h,67h, 67h,67h,67h,67h,67h
 867                          ;
 868                  stack:                          ;15 level stack
 869  0441                    end
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
ADDHLA          023D  236  506  625#
BDOS            0005   48#   49  131  902
BDOSL           0006   49#  291  297
BDOSOFF         00FE  109#  299
BIOSCONIN       003E  179#  786
BIOSCONSTAT     0038  170#  785
BIOSFUNCTIONS   FFFF   43#  147  172  181  229  773  783
BIOSIN          02F1  153  786#
BIOSTA          02EF  155  785#
BIOSTRAP        004C  176  186  197#
BREAK           0163  409  443#
BUF             0080   50#  843
CBUF            0327  505  547  831#
CBUFP           0324  498  504  575  823#
CCPFLG          00B3  107#  286  347  384
CCPLINE         0326  352  530  827#
CCPRES          0020  108#  351  388
CINF            0001   75#  214  421
CKCCP           00F8  345#  370
CKKILL          00AB  259  265#
CKSIG           009F  257#  393
CLOSEF          0010   83#  277
CONBRK          0264  433  487  661#  754
CONECH          0242  643#  746
CONIN           0196  641#  645  703
CONINF          0196  359  437  495#  641
CONINFX         0074  119#  162
CONMODE         00CF  113#  599
CONOUT          0283  647  675  677  684#  742
CONSTFX         006E  118#  159
COOKED          0300  652  806#
COUTF           0002   76#  689
CR              000D   69#  517  535  655  707  709  742  824  856
CRAWF           0006   77#  184  210  451
CREADF          000A   78#  216  419
CSTATF          000B   79#  175  208  423  445
CTL             005E   71#  677
CTLC            0003   54#  729
CTLE            0005   55#
CTLH            0008   56#  658
CTLOUT          0274  672#  727
CTLP            0010   57#  711
CTLR            0012   58#
CTLS            0013   59#
CTLU            0015   60#
CTLX            0018   61#
CTLZ            00FF   63#   65#  473  502  511  705
DCNT            00E1  117#
DELF            0013   84#  279
DIRINP          0191  431  486#
DMAAD           00D8  115#
DMAF            001A   86#  220  540
DREADF          0014   85#  545
ECHO            02FF  686  805#
ECHOC           024F  645  650#  674
ERRFLG          00AC  110#  585
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

EXCESS 03B8 194 360 850#
EXIT 007E 201 221 226# 252 267 326 468
EXITTABLE 02EB 235 775#
FALSE 0000 12# 15 828
FUNC1 0242 422 746#
FUNC10 028E 420 751#
FUNC11 0264 424 754#
FUNC6 014E 426#
GARBAGE 03BB 364 854#
GETACTIVE 03A9 260 271 844#
GETC 0196 496# 522 531
GETC1 01AE 503 510#
GETMODE 0223 336 457 597# 664
GETOUT 001B 145# 274
GFCBF 0082 93# 254
GINITF 0080 91# 248
GKILLF 0081 92# 261 266 845
GSIGF 008C 99# 258
INITTABLE 02ED 247 780#
INTERCEPT 00DD 209 211 215 217 322#
JFCBF 008F 102#
JINITF 008D 100#
JKILLF 008E 101#
KILL 000E 132# 250 282 324 368 781
LF 000A 70# 521 656 710 856
LISTCP 00D4 114# 714
LOADF 003B 89# 311
MOV7 0232 554 556 568 570 611#
MOVE 0234 615# 620
MOVEND 0324 820#
MOVSTART 02ED 779#
NBANK 000F 133#
NEXT 0009 130# 231 296 313 366 395 453 541 564 690
776
NEXTCHR 0325 515 824#
NOTC 02DC 731 735#
NOTCCP 012F 353 376#
NOTEOF 02A7 705 708#
NOTP 02C6 711 718#
OLDSTACK 03B9 331 466 475 851#
OPENF 000F 82#
PATCH 001A 136#
PATCHAREA 03DC 860#
PBUFF 0009 81# 365
PCFCBF 0086 96#
PCHRF 0005 80#
PCKILLF 0085 95#
PGDEF 00C9 112#
PGMODE 00C8 111# 345 401
PINITF 0084 94#
PLFCBF 008A 98#
PLKILLF 0089 97#
PREV 000C 131#
PROGRAM 0301 382 808#
RDECH1 02C9 724#
RDECHO 02C6 721#

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

READ	028E	693#	751					
READEN	02E0	706	709	710	738#			
READF	01D5	500	545#					
READNO	0296	702#						
READNX	0294	699#	717	737				
REALBDOS	02FC	446	801#					
REBOOTX	0216	582#	733					
REDIRECT	0133	356	389	407#				
REMOVERSX	0000	15#	294					
RESTOR	00B0	270#	392	450	513			
RESTOREBIOS	02F4	156	792#					
RESTOREMODE	02F3	150	789#					
RETMON	0186	415	471#					
RNAME	0010	134#						
RSXF	003C	90#	218	394				
RSXFUNC	008A	219	245#					
RUBOUT	007F	67#						
SAVAREA	03AA	550	847#					
SCBA	003A	106#						
SCBADR	02ED	158	285	549	584	598	714	781#
SCBF	0031	88#						
SERIAL	0000	126#						
SETDMA	01D0	540#	548	573				
SKIP	0175	338	379	396	455#	474		
SKIP1	017F	441	466#					
SPACE	0018	135#						
STACK	0441	332	868#					
STATFLG	02EA	212	434	662	763#			
SUBCR	0323	818#						
SUBDOS	01D7	278	280	546#				
SUBFCB	0303	255	811#					
SUBMIT	FFFF	14#	62	276	340	526	826	853
SUBMOD	0311	815#						
SUBRC	0312	816#						
SUBUSR	0302	559	810#					
TAB	0009	68#	657					
TRAP	0046	129	191#					
TRAPJMP	0006	128#	202					
TRUE	FFFF	11#	14	43	363			
UDMA	03A7	223	571	843#				
USERF	0020	87#						
USRCODE	00E0	116#	555	560	569			
WBOOT	0000	47#	374	589				

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_ _____

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____ ____